AD NUMBER

AD850054

NEW LIMITATION CHANGE

TO

Approved for public release, distribution unlimited

FROM

Distribution authorized to U.S. Gov't. agencies and their contractors; Administrative/Operational Use; Mar 1969. Other requests shall be referred to Comptroller of the Army, Attn: COMPT-CA[R], Washington, DC 20310.

AUTHORITY

COMPT-CA ltr, 5 Nov 1974

RESEARCH
ANALYSIS
CORPORATION

A Monte Carlo Simulation Approach to Cost-Uncertainty Analysis

ECONOMICS AND COSTING DEPARTMENT
TECHNICAL PAPER RAC-TP-349
Published March 1969

A Monte Carlo Simulation Approach to Cost-Uncertainty Analysis

by
Donald F. Schaefer
Frank J. Husic
Michael F. Gutowski

RAC Research Analysis Corporation
McLean, Virginia

FOREWORD

A technique is described for preserving in cost estimates more of the information available to the engineers and cost analysts participating in the process of estimating. The added information, principally in the form of a frequency distribution for system costs, should prove useful to the persons faced with the problem of using the cost estimate in the decision-making process.

An actual test of the technique, made with a major component of the Main Battle Tank 70, will be documented in a subsequent publication.

Arnold Proschan
Head, Economics and Costing Department

CONTENTS

BLANK PAGE

A Monte Carlo Simulation Approach to Cost-Uncertainty Analysis

RAC

ABSTRACT

An important aspect of cost research is the measurement of the uncertainty inherent in the projection of system cost. Approaches to this problem have in the past focused on the decision maker's intuition or on sensitivity analysis. Only recently have approaches utilizing such tools as statistical decision theory and probability theory been formulated.

This study focuses on the Monte Carlo simulation approach to uncertainty in cost analysis. This approach requires (a) expression of input estimates as probability distributions reflecting uncertainty and (b) cost equations pertinent to a particular model.

The Monte Carlo simulation approach then generates (a) the frequency distribution for system cost and (b) statistical measures that illustrate the nature and magnitude of system cost uncertainty.

Two models are developed, the Beta model and the Weibull model, each of which reflects a particular distribution form for the inputs. The relative costs and advantages of each model are compared.

A user's guide to the program and complete program listings are presented in App A.

INTRODUCTION

The purpose of this paper is to describe a technique for quantifying uncertainty in cost analysis. Approaches to the measurement of the uncertainty present in cost models have in the past generally centered on the intuition of the decision maker or, at best, on sensitivity analysis. Only recently have approaches based on statistical decision theory and probability theory been attempted.

The technique of uncertainty analysis is defined for purposes of this paper as the use of probability distributions as inputs to aggregate models. This analysis can be used in several ways. For example, it can be valuable where there is a major lack of information concerning the value of input parameters. This can be true of systems not yet in existence or of estimates of historical parameters for which data are fragmentary. It can also be of value in estimating the cost of an existing system for which actual data are available. Samples of these data can be used to construct distributions for input variables.

The model developed is called the "Monte Carlo Uncertainty Analysis Model." It is to be used with any cost model. The model allows the user to specify probability distributions for the cost model input variables rather than the usual single-point estimates. These are then input to the cost model in the uncertainty analysis model, and the outputs are probability distributions and confidence intervals rather than single-point estimates of costs. There are two variations of the model, differing as to types of probability distributions that can be used to characterize inputs. One assumes a Beta distribution and the other a Weibull. The availability of two types of distribution permits the user more flexibility in specification. The rationale for use of one or the other approach rests in which type of distribution the analyst feels most closely approximates the distribution of the input variables of the particular problem and which set of input specifications he can most easily and accurately provide.

The first chapter of the paper discusses the value of uncertainty analysis models in general and describes the basic methodology of techniques developed. The second chapter discusses actual use of the techniques in several simplified examples.

The appendix provides the user with the information necessary to effectively implement the programs. The precise format of user-provided inputs is described. Error messages are incorporated into the program to ensure conformity with input requirements. A description of program logic is followed by a complete listing of the Weibull and Beta programs.

Chapter 1

COST-UNCERTAINTY ANALYSIS: AN OVERVIEW

COST UNCERTAINTY IN DECISION MAKING

The process of decision making and thus the fundamental task of the decision maker is to choose among alternative courses of action. Often such choice will involve a cost-benefit analysis of either a formal or an informal nature. Within the context of a formal analysis, what has been given the decision maker as costs may now be examined.

The precise calculation of costs is not a difficult task. The availability of computers and cost models can reduce this to routine. The calculation itself is precise and can be done rapidly in minute detail.

The inputs to these precise calculations are, however, not precise; in fact they are often quite the opposite. The errors present in each input are passed on to various aggregations until a total cost is arrived at that somehow reflects each individual error. Thus, cost data inputs are combined in a computer model with a multiplicity of equations and hundreds of other inputs to form a single estimate of total costs. This single estimate is presented to the decision maker with the implication that, although it may not be perfect, it is certainly the best available estimate and without any statement as to likelihood of occurrence or range of other possible values.

But in the generation of this aggregate number, hundreds of imprecise numbers may have been used. This fact is not emphasized to the decision maker nor does he have a basis on which to judge the precision of the numbers. Cost models as currently conceived and used, then, uniformly withhold from the decision maker some information that might be vital to his decision. He has been denied some available information on the accuracy of the estimates.

As an example, suppose that one of the elements to be costed is a missile airframe. The missile is not yet designed or built, so no production-cost data are available. A cost-estimating relation (CER) could be used, for example, to estimate that the airframe cost for a new missile would be about $32,000. The design group, however, warns that some added sophistication might make the CER predict on the low side, but the improvements in some manufacturing techniques promise lower costs. Some quick calculations show estimates as low as $26,000 and as high as $45,000. Each of these calculations is based on a set of assumptions concerning labor and overhead rates, material costs in the future, and design details not yet firm and subject to some uncertainty.

For each of a multitude of other important inputs, similar assumptions are made, and a single aggregate cost is obtained as the output of the cost model. The hypothetical decision maker is then forced to make his final decision on a single value derived from several imprecise input values. If he can be provided with some additional knowledge concerning the impact of the uncertain variable, a more rational decision can be made.

The dangers of the single-cost estimate have been recognized for years, and several strategies have been developed for augmenting the analysis or circumventing the difficulty. One is isolation of the differences between alternatives. Cost elements common to alternatives are estimated in a similar or identical manner so that only the uncertainties of the unique feature of alternatives affect relative cost. Another is the use of sensitivity analysis. In sensitivity analysis the impact of errors in estimates and assumptions is computed, and error sources important to the choice of alternatives are identified. Such an analysis can produce proof in insensitivity, or it can provide evidence that, within a relevant range of values, choice is or is not affected by estimation error.

When judgments about the relevant range of all variables in a model are considered, sensitivity analysis produces an array of numbers that includes the analysts' beliefs concerning the limits of the variables but excludes any knowledge of their relative probability. No probability statements are furnished, and the decision maker could be led to believe that all numbers in the array are equally likely.

The choice of a relevant range of values for sensitivity analysis is both difficult and critical to its usefulness. It also reveals a dilemma inherent in the application of sensitivity analysis. If the analyst has evidence that the value of one of the inputs is constrained within some upper and lower limits, then this same evidence may provide information on the relative likelihood of particular values. In some cases an intuitive belief about the range may provide him with equally valid suppositions concerning probabilities of the values within the range.

In the previous example of the missile cost, if the reasoning that fixed the relevant range of costs at $26,000 to $45,000 could yield information on the probability of occurrence, the missing information could be supplied in a form useful to the decision. A possible statement might be that there is a 99 percent probability that the cost is less than $42,000.

When all uncertain elements of a given cost model are combined, sensitivity analysis may, for example, produce a maximum cost of $810 million even though there is only one chance in a hundred that costs will exceed $700 million. In the sensitivity analyses, if the $800 million number exceeded slightly the cost of another alternative under similar sensitivity assumptions, the decision maker may have been furnished an unlikely cost set for consideration along with all other sensitivity sets, and this set may uniquely favor a different alternative. Knowing just a few simple probability statements, then, can make unnecessary the time-consuming consideration of cost estimates whose likelihood is very remote.

In short, although sensitivity analysis is a powerful tool for portraying the results of estimating error, it may lead to considering highly unlikely

situations. The same reasoning that leads to a determination of relevant range for sensitivity analysis has the potential of providing key information in decision making.

Value of Probability Information

The use of probability distributions as inputs to aggregate models has recently captured the attention of the cost analysts. The two most widely used techniques for handling subjective probability information are the derivation of moments[1] and the Monte Carlo simulation. This paper focuses on the Monte Carlo simulation approach and relies heavily on the works of David B. Hertz,[2] Paul F. Dienemann,[3] and W. D. Lamb.[4]

A discussion of the Monte Carlo m dology is preceded by several examples designed to illustrate the potential value of probability distribution information to the decision maker. In each illustration the frequency distributions for two alternatives are shown. The horizontal axis in each case represents the cost of the alternative and is increasing to the right. The vertical axis represents the likelihood of occurrence at each cost level. Each of these is a hypothetical case in which equal effectiveness or other benefits are assumed. The decision maker's problem is that of choosing the least-cost alternative. If only single-point cost estimates were provided the decision maker would of course feel constrained to select the lower cost in each case. However, Fig. 1 demonstrates how information provided by probability estimation could modify his outlook.

The peak of each curve is at the most likely, or modal, value and that is, in these hypothetical cases, the only cost total that would be furnished a decision maker in the absence of uncertainty analysis. In the example in Fig. 1 B is expected to be less expensive but it has a much larger variance, so that extremely high costs are more likely than with A. Faced with this dilemma, the decision maker may decide to avoid extreme costs by choosing A or to gamble on the expected lower costs of B. The decision cannot be prescribed for him, but probability distributions can give him awareness of a pitfall in selecting the "less expensive" alternative.

Figure 2 illustrates a clear-cut case. The largest possible cost of one alternative is less than the smallest possible cost of the other. Decision makers furnished with this information are not likely to choose differently than if only point estimates were given.

In Fig. 3 a single-point estimate would furnish no basis for choice. With most likely values essentially the same the decision maker must look elsewhere for (ifferences. If the probability distributions are furnished, however, it is apparent that the costs cannot be regarded as equal. If they appear as shown here it is presumed that A would be chosen, since it can be better accommodated in the budgeting and financial management system. It is possible that much-lower-than-mean costs may have some value too, and as a result it becomes impossible to speculate on choice outside specific cases. The important thing is that information helpful in the decision process has been furnished that would not otherwise be available.

The costs associated with two alternatives may or may not be significantly different. Figure 4 illustrates two cost totals, each with the same

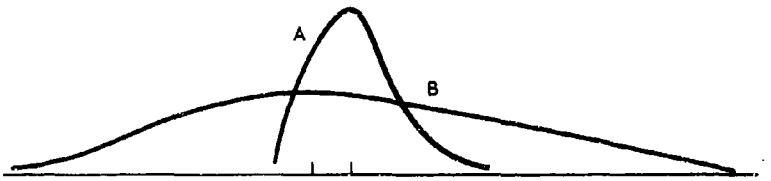


Fig. 1—Alternative Comparison, Case 1

Most likely value A > most likely value B
variance A < variance B

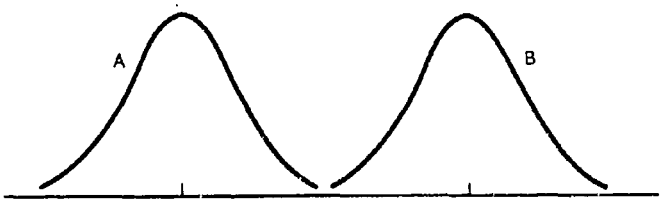


Fig. 2—Alternative Comparison, Case 2

Most likely value A < most likely value B
No overlap of distributions

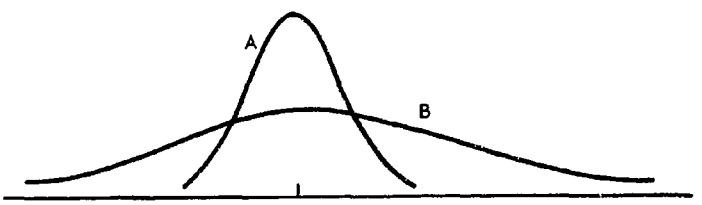


Fig. 3—Alternative Comparison, Case 3

Most likely value A = most likely value B
Variance A < variance B

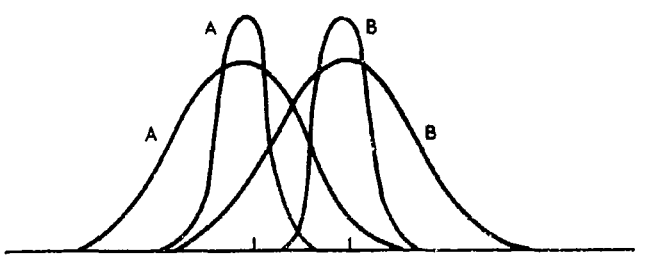


Fig. 4—Alternative Comparison, Case 4

Most likely value A < most likely value B
Variance A = variance B; most likely value A < most likely value B
Variance A′ = variance B′; variance A′ < variance A
Variance B′ < variance B

RAC

variance. If variance is low A′ and B′, the situation is as in Case 2; costs are different and offer a basis for choice as conclusive as the magnitude of the difference might indicate. Larger variances, A and B, diminish the importance of the most-likely-value cost difference. At some combination of "closeness" and high variance the cost difference may not be of significance in the selection of an alternative. In any case, quantification of the probabilities of the differences is a useful contribution to the decision maker's understanding.

These four examples, it should be noted, differ only quantitatively. They are offered in these forms to illustrate the range of possibilities in which useful information may come from a knowledge of the probability distributions of total cost.

METHODOLOGY

This section presents the basic methodology of two variations of the Monte Carlo model. One assumes that uncertain inputs can be characterized by Beta distributions. The other variation uses the Weibull distribution.

Overview of Logic

Since the two variations of the Monte Carlo technique are identical as to logic and statistical methods, the general logic of both models can be discussed together. Basically the models consist of a Monte Carlo simulation, a listing of the simulation results by class intervals, and a plot of the output distributions. In the simulation the value of each input parameter is chosen randomly, based on the probability distribution of the parameter. When this procedure is repeated many times, a range of values is produced. These values are aggregated into a number of class intervals and results are plotted. Finally, since the results of the simulation produce a very crude plot, a smoothing routine produces a more regular version of the output distribution.

Beta Variant

This approach is quite similar to one formulated by P. Dienemann of The RAND Corporation.[3] It assumes that cost distributions can be approximated by Beta distributions. These are unimodal and continuous finite at both upper and lower bounds and can be skewed or symmetric.*,[5] This section discusses (a) required inputs, (b) general logic of each technique, and (c) outputs.

Inputs. To specify the Beta distribution for an input variable the following input parameters are required:

(1) XP (the most probable value of the distribution): If one views the probability distribution of an X, Y plane, the most probable value corresponds to the point having the greatest Y value. For example, the points P and P_1 are the most probable values for the distributions shown in Fig. 5. In the figure the abscissa describes the values that the input variable may assume and the

*A more comprehensive discussion of this type of distribution can be found in Mood and Graybill,[5] p 129.

ordinate gives the likelihood of occurrence of each of those values. In the symmetric distribution (curve A of Fig. 5) the most probable, or modal, value as it is generally known also corresponds to the expected or mean value.*

(2) XH (high value): This is the point to the right of the modal value where the probability distribution meets the abscissa given the X values increase from left to right. In Fig. 5 the points H and H_1 are the high values.

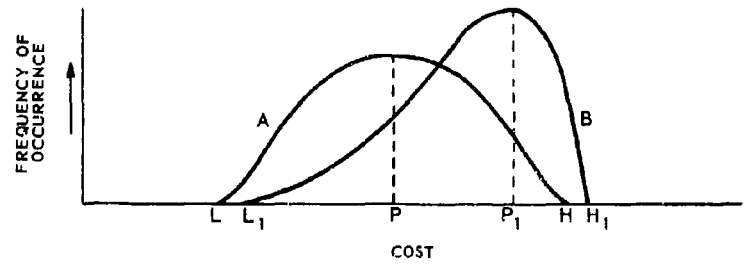


Fig. 5—Typical Beta Distributions

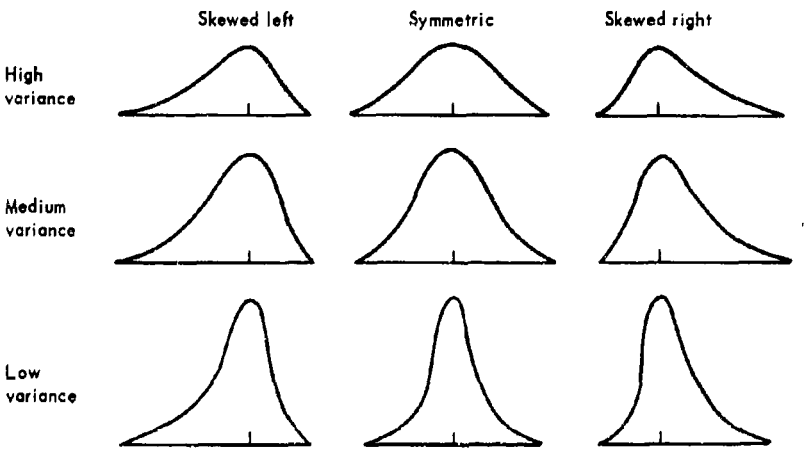


Fig. 6—Sample Set of Distributions

(3) XL (low value): This is the point to the left of the modal value where the probability distribution meets the abscissa given the X values increase from left to right. In Fig. 5 the points L and L_1 are the low values. In Fig. 5 they are the zero point of a standard X, Y coordinate.

(4) Distribution Type: Finally the user must compare his concept of the probability distribution of the input variable with a set of standard distributions and choose the one that most closely approximates his concept. The set of distributions now being used is shown in Fig. 6. These distributions can be described qualitatively on the basis of (a) being symmetric or skewed or (b) having different degrees of variances. This is illustrated in Fig. 6. Most un-

*For a strict definition of this see Mood and Graybill,[5] p 103.

certain inputs should be approximated reasonably by one of the nine distribution types.

(5) The Cost Model: The user must provide the costing equations that compose his cost model. These are embedded within the uncertainty analysis model and are necessary to compute the values from which the final plot of the distributions is to be made. To illustrate with an extremely simple example, suppose one were interested in calculating the probability distribution of personnel and maintenance costs for an aircraft battalion. Further, suppose personnel costs were available on a per capita basis and maintenance on a per aircraft basis, then the cost model submitted by the user might look like this:

F1 = number of men x dollars per man

F2 = number of aircraft x maintenance cost per aircraft

Total cost = F1 + F2

The five types of input described above must be prepared for each input variable in the cost model. The necessary model inputs having been discussed, the general logic of the model may now be considered.

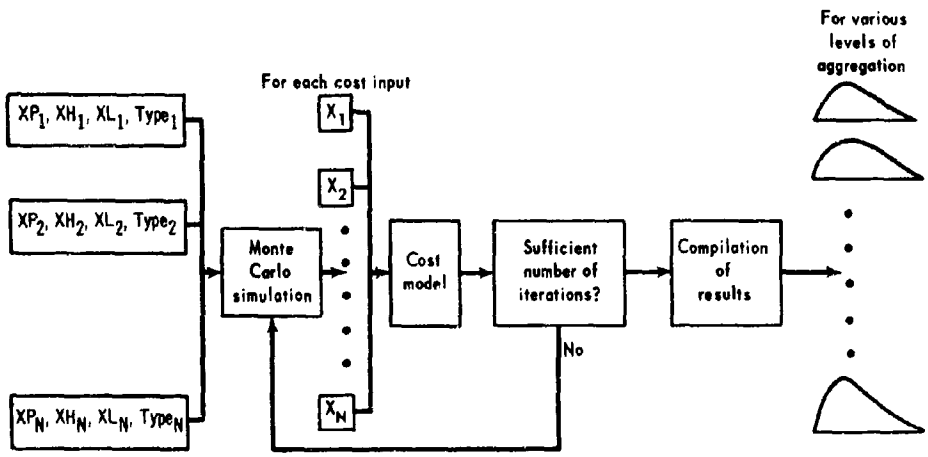


Fig. 7—Monte Carlo Technique, Beta Variant

Logic and Output. Figure 7 illustrates the basic methodology of the model. For each of the uncertain costs inputs, the four required parameters are provided by the user. Using this information, the model randomly selects a single value for each input. This particular set of values is then input to the cost model that is now embedded in the uncertainty model. This produces a particular set of output values that are stored for use later in building the final distribution curves. The process is repeated 1000 times, which should be a sufficient number of iterations to develop an accurate total cost distribution. The number of iterations can be changed easily to meet user requirements. If the specified number of iterations has not been run, the model will recycle to again (a) randomly select different values for each input, (b) compute output values for these selected inputs using the embedded cost equations, and (c) store these results for further use.

When the specified number of iterations has been run, the model moves to a new phase. This involves compilation of the results of e ch run to (a)

prepare frequency distributions for aggregate cost and (b) compute statistical measures such as mean and variance. First, the values are ordered into class intervals covering the range of possible values. The frequency diagram produced by this ordering is then smoothed to produce the more familiar form of probability distribution illustrated in Fig. 7. The next chapter contains a more detailed description of the actual form of output generated by the model.

Weibull Variant

In this approach inputs are characterized by a Weibull distribution.*,[6,7] Weibull distributions, like Beta distributions, are unimodal continuous and can be symmetric or skewed but, unlike Beta distributions, are infinite at both upper and lower bounds. Since the Weibull variant approach is identical in logic and output to the Beta variant just described, only the input requirements of the Weibull variant will be discussed in detail.

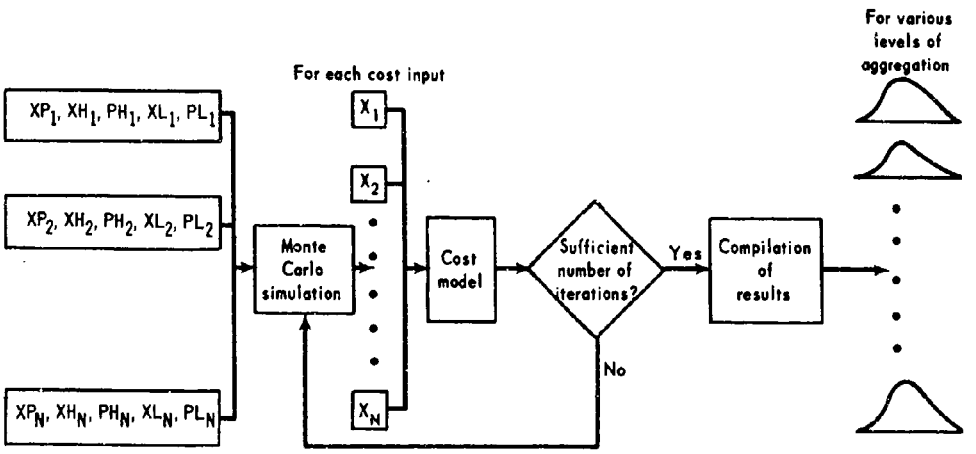


Fig. 8—Monte Carlo Technique, Weibull Variant

Inputs. Figure 8 illustrates the methodology of the Weibull variant. As illustrated in the figure the input requirements for each cost input are:

(1) XP (most probable value): This has the same meaning as the most probable value input for the Beta variant. The discussion is identical and therefore will not be repeated at this point.

(2) XH and PH: Since these inputs are interrelated, both can be discussed in one section. XH is a value of the input that is higher, i.e., greater than, the most probable value specified for the input. PH is the probability that the value of that input will be some quantity greater than XH. Figure 9 illustrates this graphically. The X axis is the value of the input variable and the Y axis is the probability of occurrence of different X values. Upon closer inspection, it is clear that this input specification is analogous to the specification of XH for the Beta variant. In the case of the Beta variant the user was asked to specify XH such that the probability of the input being greater than

*A complete mathematical description of the Weibull distribution can be found in Refs 6 and 7.

XH was zero. Thus PH is zero. Since the Weibull distribution is infinite and thus has no fixed bound the user is asked to specify XH and PH in such a way that the probability of the value being greater than XH is PH.

(3) XL and PL: These are analogous to XH and PH where XL is some value less than XP such that the probability of the input variable being less than XL is PL.

Logic and Outputs. With these inputs the Weibull variant operates in essentially the same fashion as the Beta variant. Values are randomly selected for each of the cost inputs, and these are processed through the cost model with results being stored for development of the final distributions. After the specified number of iterations is completed the model orders the results of each iteration and develops frequency distributions, which are smoothed and then finally graphed.

Model Limitations. The techniques developed in this paper provide many advantages over more conventional forms of cost analysis. Uncertainty about input variables can be explicitly described. The simultaneous interaction of uncertainties in many variables can be assessed and then graphically displayed

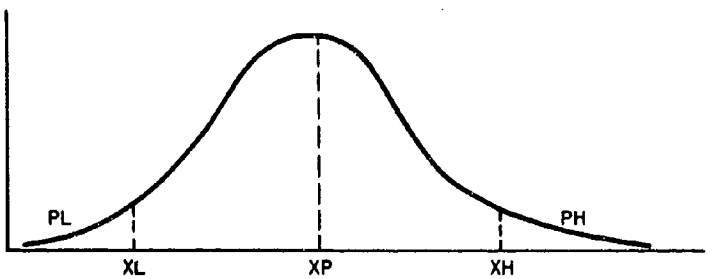


Fig. 9—Weibull Distribution

to those confronted by decision. The sensitivity of final results to each or a set of inputs can be efficiently tested. However, it would be at best naïve to claim there are no disadvantages to use of these techniques. Some major considerations are:

(a) The assumption of Beta or Weibull distribution may not be a sound one.

(b) Much additional input information is necessary to operate the model.

(c) The model assumes independence of the input variables. This can prove limiting in certain situations. There are at present several approaches to handling dependency. Among these are incorporation in the model of the functional relation between the variables, statement of the dependent variable in terms of auxiliary variables, and use of joint probability distributions. Though these can eliminate dependence in a majority of costing problems, the systematic handling of input dependency remains a very open area for research.

Chapter 2

IMPLEMENTATION OF MONTE CARLO MODELS

A thorough understanding of the input requirements and output interpretation of uncertainty analysis models is necessary before attempting to apply the models to specific problems. The user of the models should be aware of model limitations, sources of input, output interpretation, and the differences between the Beta and Weibull models in order to implement the models effectively. A discussion of computer time requirements for the models is presented to acquaint the potential user with the anticipated costs of applying the program to a given problem. Format of inputs, computer program logic, and complete program listings are presented in the appendix.

SOURCES OF INPUT INFORMATION

The Beta and Weibull simulation models are designed to handle any type of cost function. Information concerning uncertain variables may come from a variety of sources. The user of the model must be prepared to obtain the precise input information required by the model from these diverse sources.

In many cases the best source of information about an uncertain input is a person or agency familiar with the variable in question and possible sources of variation. For example, a certain cost model may require the cost of a newly designed equipment item as an input. An engineer who is working on the design of the equipment and who is acquainted with the possibilities of production delays and future design modifications may be the best source for necessary inputs. The input is still an estimate, but by minds more intimately informed on the details of a specific variable.

Another example is a cost model that requires manpower level as an input. The best source of information may be an experienced military advisor who is familiar with manpower requirements under varying circumstances. In both cases, the expert providing the information may be unfamiliar with statistics and the precise meaning of model input requirements such as finite upper and lower limits or assignment of probabilities.

It is necessary for the user to precisely define the variable that must be estimated and to carefully delineate the information needed from such expert sources of information. An excellent example of techniques for obtaining program inputs from such sources is presented in "A Technique for Probability Assignment in Decision Analysis" by W. D. Lamb.[1]

Information concerning uncertain inputs is frequently derived from statistical techniques such as regression and correlation analysis. Examples are (a) cost-estimating relations that express the cost of equipment or material as a function of design or performance characteristics, (b) overhaul rates for military vehicles, and (c) the ratio of draftees to total accessions to the armed services. Information obtained from regression models can be readily translated into the form required by the uncertainty analysis models. Whether the best information source is an engineering expert, a regression model, or any combination of sources, the information can be expressed in the form required for an input in the uncertainty model.

MODEL INPUT REQUIREMENTS

The Beta Model

The necessary inputs for the Beta model are:

(1) XP: The most likely value of the variable
(2) XH: A finite upper limit
(3) XL: A finite lower limit
(4) Distribution type*

The first three inputs must be obtained from the best available source; the fourth can be derived from the first three by the user. The most readily available input should be XP, the most likely value of the variable. Obtaining finite upper and lower limits presents a more serious problem.

In many cases the expert source of information will hesitate to provide a finite upper limit for an uncertain variable, for to do so would be to imply that the value of the variable cannot possibly exceed this limit. Consider the case of the engineer estimating the cost of an advanced-design engine at an early stage in its development. The remote possibility of a major change in end-item design characteristics or an insoluble technical problem could drive the cost up considerably. These possibilities may preclude determination of a finite upper limit or require an extremely high upper limit to account for any feasible situation. Similar problems could be encountered by the military analyst who is predicting troop strength in some future time period. The probability, though very small, of an armed conflict requiring massive troop build-up or of a technological breakthrough that will replace existing manpower requirements dramatically must be considered in establishing the limits of the distribution.

The finite nature of the Beta distribution implies that the value of the uncertain variable must remain within the range described by the upper and lower limits with 100 percent certainty. The distribution of an uncertain variable obtained through regression analysis with normal error terms has infinite range. The model user can, however, use a 99 percent confidence interval as a reasonable proxy for certainty.

Once the most likely, high, and low values are determined, the model user can derive the distribution type. If the difference between the upper limit and

*Distribution types characterized by direction of skewness and relative variance are presented in Fig. 8.

the most likely value (XH - XP) equals the difference between the most likely value and the lower limit (XP - XL), the distribution is symmetric. If (XP - XL) is greater than (XH - XP), the distribution is skewed to the left. If (XH - XP) is greater than (XP - XL), the distribution is skewed to the right. Once the direction of skewness is determined, the user can ask the expert source to choose from the three figures representing the relative variance of distributions in the category. If the information is obtained from a regression model, the symmetric medium variance curve is the most appropriate since it approximates the transformation of a normal curve to a Beta distribution.

The Weibull Model

Weibull input requirements are:

(1) XP: The most likely value
(2) XH: The upper limit
(3) PH: Probability that the value of the input will exceed XH
(4) XL: The lower limit
(5) PL: Probability that the value of the input will be less than XL

Values for XP, XL, and XH are obtained as in the preceding model. The major difference is that XL and XH are not finite limits. Some of the problems associated with the Beta model are thus eliminated. The expert source may be more willing and able to present bounds with a probability value attached to them. When uncertain inputs are obtained by means of regression analysis with error terms normally distributed, the Weibull input requirements are easily and accurately met.* Since the variable has a known normal distribution, XP is the predicted value of the variable and PH and PL are chosen by the user to determine XL and XH. If the user chose 5 percent as values for both PH and PL, a 90 percent confidence interval about the predicted value would yield XL and XH.

OUTPUT FORMAT

The outputs of the Beta and Weibull Monte Carlo programs are identical in format. Both routines produce two plots of the total cost distribution and a table of frequencies within class intervals.

The first plot produced by the program shown in Fig. 10 is that of the total cost distribution as produced by the Weibull Monte Carlo simulation. The Y axis, running horizontally across the top of the computer output page, measures the number of total cost estimates lying within a given class interval. The X axis, running vertically along the left-hand side of the computer output page, measures total cost. The "Xs" that form the distribution show how many times total cost fell within each interval. Four other values, XMIN, XMEAN, XMAX, and STD DEV, are printed. These represent the minimum total cost, the mean total cost, the maximum total cost, and the standard deviation of the total cost distribution. In Fig. 10, for example, the minimum cost is $98,422, the mean cost is $116,111, the maximum cost is $131,859, and the standard deviation is $5697.

*"The Weibull density function is nearly symmetric (approximating the normal distribution) when its shape parameter is approximately 3.5." See Lamb,[4] p 15.

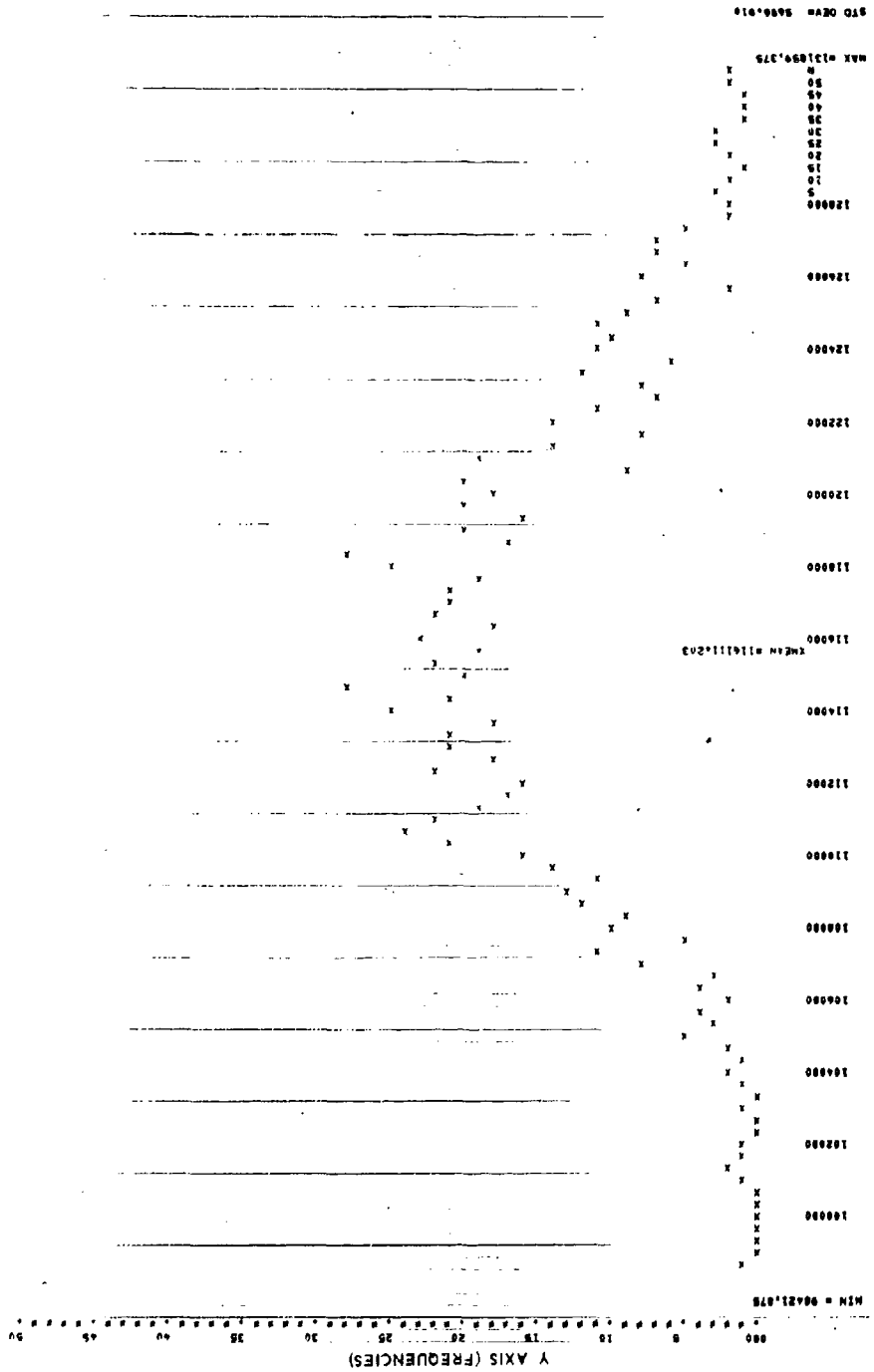


Fig. 10—Unsmoothed Output

The second plot, Fig. 11, is a smoothed version of the first. The cost data generated by the Monte Carlo simulation are smoothed by fitting them to a polynomial by the method of "least squares."* The frequencies are rounded to the nearest integer.

The table of frequencies produced by the program and shown in Fig. 12 has four columns of information. The leftmost column contains the midpoint of each of the 100 class intervals. Column 2 contains the number of total cost estimates, produced by the Monte Carlo simulation routine, that lie within each cost interval. The third column contains the Y value produced by the smoothing routine for each class interval. This value is rounded to the nearest integer. The fourth column is the difference between the second and third columns. In Fig. 12, for example, the highest class interval has a midpoint of $131,692, a frequency of 2.0, a smoothed frequency of 2.4291, and the difference between the two frequencies is 0.4291.

COMPUTER TIME REQUIREMENTS

The purpose of this section is to provide the user with computer time requirements as a measure of cost to enable him to evaluate these programs in terms of cost effectiveness.

For the Beta and the Weibull programs two parameters were varied in order to determine the sensitivity of computer time consumption. These parameters are the number of iterations in the Monte Carlo simulation and the number of variables. It was considered a priori that these two parameters would explain most of the time variations from run to run.

The number of iterations was set at 500 and 1000 for the purposes of experimentation. The number of cost variables was changed from 3 to 15. Along with the change in the number of variables there was a change in the complexity of the algebraic operations. These two cost models are shown in Fig. 13.

Timing results for the Beta and Weibull programs are shown in Table 1. Total time results include execution time, compiler time, system time, and load time. Since execution time varies with the number of iterations and the compiler, load, and system times are roughly fixed, it is also useful to see the execution times alone since they are roughly analogous to recurring costs.

A number of points should be noted in interpreting Table 1. First, the times shown are for an IBM 7044. If the programs are used on a different computer the times will vary. Second, the total times are quite small, never exceeding 5 min. Third, for the most part execution time is less than 50 percent of total time. Finally, the execution time will not vary greatly with the number of variables since the major difference is the number of data cards that must be read by the computer.

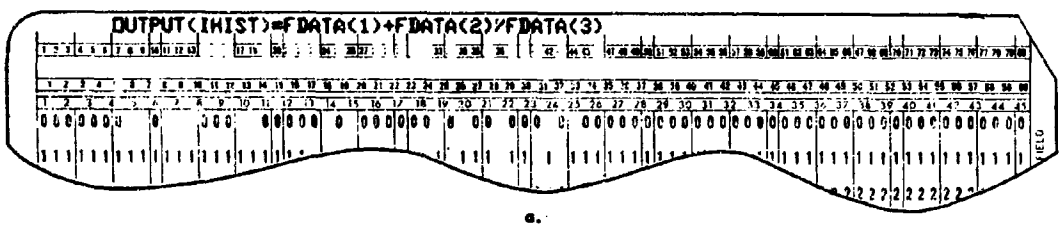
From this information it seems probable that the vast majority of cost models will require less than 10 min of computer time.

*The method of least squares chooses the coefficient of the fitted polynomial so that the sum of the squared deviations between the polynomial and real data is minimized.

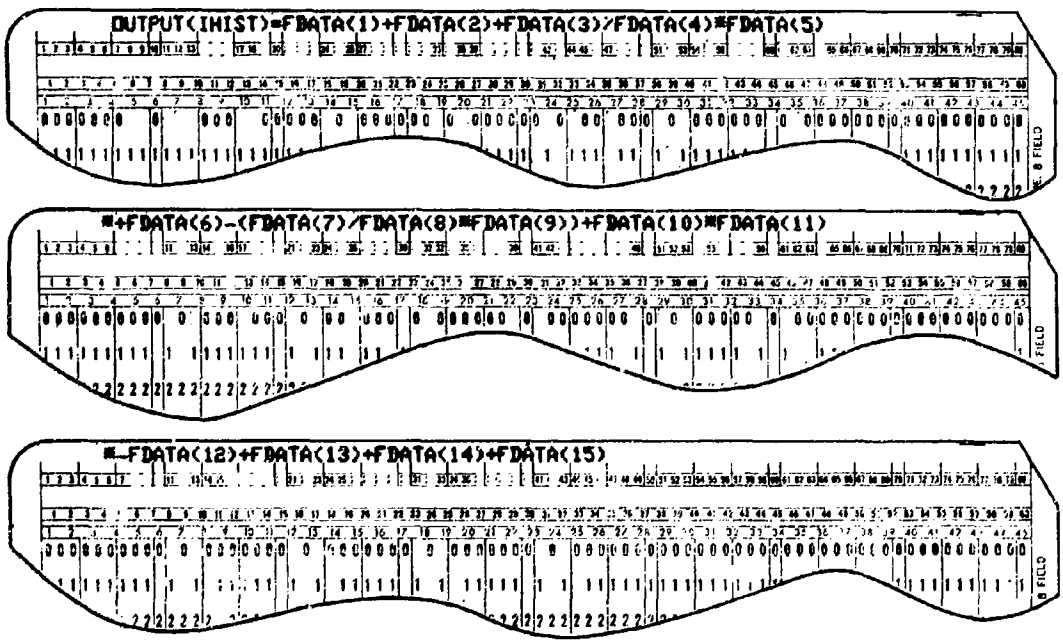
Fig. 11—Smoothed Output

COST INTERVAL	FREQ	SMOOTHED FREQ	RES
88589.6625	1.0000	1.0814	-.0814
90923.4375	0.0000	.1174	-.1174
92257.2125	0.0000	-.2453	.2453
93592.1875	0.0000	-.2410	.2410
94926.9625	0.0000	-.0400	.0400
100260.4375	0.0000	.2389	-.2389
100595.2125	0.0000	.5179	-.5179
100929.6875	1.0000	.7509	.2491
101264.0625	2.0000	.9160	1.0840
101598.4375	1.0000	1.0093	-.0093
101932.8125	1.0000	1.0395	-.0395
102267.1875	0.0000	1.0238	-1.0238
102601.9625	0.0000	.0841	-.0841
102935.9375	1.0000	.9449	.0551
103270.3125	0.0000	.9306	-.9306
103604.6875	1.0000	.9642	.0358
103939.0625	2.0000	1.0660	.9340
104273.4375	1.0000	1.2529	-.2529
104607.8125	2.0000	1.5376	.4624
104942.1875	5.0000	1.9297	3.0703
105276.5625	3.0000	2.4329	.5671
105610.9375	4.0000	3.0479	.9521
105945.3125	2.0000	3.7710	-1.7710
106279.6875	4.0000	4.5953	-.5953
106614.0625	3.0000	5.5104	-2.5104
106948.4375	8.0000	6.5034	1.4966
107282.8125	11.0000	7.5591	3.4409
107617.1875	5.0000	8.6611	-3.6611
107951.9625	10.0000	9.7915	.2085
108285.9375	9.0000	10.9324	-1.9324
108620.3125	12.0000	12.0657	-.0657
108954.6875	13.0000	13.1743	-.1743
109289.0625	11.0000	14.2422	-3.2422
109623.4375	14.0000	15.2552	-1.2552
109957.8125	16.0000	16.2010	-.2010
110292.1875	21.0000	17.0700	3.9300
110626.9625	24.0000	17.8551	6.1449
110960.9375	22.0000	18.5523	3.4477
111295.3125	19.0000	19.1602	-.1602
111629.6875	17.0000	19.6804	-2.6804
111964.0625	16.0000	20.1168	-4.1168
112298.4375	22.0000	20.4760	1.5240
112632.8125	18.0000	20.7662	-2.7662
112967.1875	21.0000	20.9968	.0032
113301.5625	21.0000	21.1783	-.1783
113635.9375	18.0000	21.3209	-3.3209
113970.3125	25.0000	21.4344	3.5656
114304.6875	21.0000	21.5276	-.5276
114639.0625	28.0000	21.6069	6.3931
114973.4375	20.0000	21.6769	-1.6769
115307.8125	22.0000	21.7390	.2610
115642.1875	19.0000	21.7919	-2.7919
115976.5625	23.0000	21.8309	1.1691
116310.9375	18.0000	21.8486	-3.8486
116645.3125	22.0000	21.8349	.1651
116979.6875	21.0000	21.7777	-.7777
117314.0625	21.0000	21.6636	-.6636
117648.4375	19.0000	21.4788	-2.4788
117982.8125	25.0000	21.2103	3.7897
118317.1875	28.0000	20.8469	7.1531
118651.5625	17.0000	20.3803	-3.3803
118985.9375	20.0000	19.8063	.1937
119320.3125	16.0000	19.1253	-3.1253
119654.6875	20.0000	18.3434	1.6566
119989.0625	18.0000	17.4724	.5276
120323.4375	20.0000	16.5297	3.4703
120657.8125	9.0000	15.5377	-6.5377
120992.1875	19.0000	14.5232	4.4768
121326.5625	14.0000	13.5154	.4846
121660.9375	8.0000	12.5444	-4.5444
121995.3125	14.0000	11.6393	2.3607
122329.6875	11.0000	10.8252	.1748
122664.0625	7.0000	10.1217	-3.1217
122998.4375	8.0000	9.5401	-1.5401
123332.8125	12.0000	9.0823	2.9177
123667.1875	6.0000	8.7396	-2.7396
124001.5625	11.0000	8.4925	2.5075
124335.9375	10.0000	8.3119	1.6881
124670.3125	11.0000	8.1607	2.8393
125004.6875	9.0000	7.9975	1.0025
125339.0625	7.0000	7.7806	-.7806
125673.4375	2.0000	7.4731	-5.4731
126007.8125	8.0000	7.0481	.9519
126342.1875	2.0000	6.4939	-4.4939
126676.5625	7.0000	5.8177	1.1823
127010.9375	7.0000	5.0483	1.9517
127345.3125	5.0000	4.2348	.7652
127679.6875	2.0000	3.4425	-1.4425
128014.0625	2.0000	2.7450	-.7450
128348.4375	3.0000	2.2114	.7886
128682.8125	2.0000	1.8907	.1093
129017.1875	1.0000	1.7950	-.7950
129351.9625	2.0000	1.8852	.1148
129685.9375	3.0000	2.0635	.9365
130020.3125	3.0000	2.1841	.8159
130354.6875	1.0000	2.0889	-1.0889
130689.0625	1.0000	1.6912	-.6912
131023.4375	1.0000	1.1118	-.1118
131357.8125	2.0000	.9116	1.0884
131692.1875	2.0000	2.4291	-.4291

Fig. 12—Table of Frequencies Plotted in Figs. 10 and 11



a.



b.

Fig. 13—Cost Models a and b Used in Generating Timing Requirements

TABLE 1

Computer Time Requirements

Program	Total time,[a] min		Execution time	
	V = 3[b]	V = 15	V = 3	V = 15
Beta				
I[c] = 500	2.64	3.07	0.64	0.87
I = 1000	3.17	3.30	0.74	1.21
Weibull				
I = 500	2.39	2.67	0.25	0.44
I = 1000	2.49	2.92	0.35	0.69

[a]Total time includes compiler time, system time, load time, and execution time. All times are measured on an IBM 7044 computer.

[b]V = number of cost variables.

[c]I = number of iterations.

CONCLUSION

The Monte Carlo simulation models provide a tool for expressing uncertainty in item cost estimates and deriving a corresponding expression of uncertainty in total costs. The user must, however, be aware of the potential, the costs, and the limitations of the models before applying them to solve a particular problem.

Appendix A

USER'S GUIDE TO BETA AND WEIBULL PROGRAMS

WEIBULL INPUT REQUIREMENTS

The user of the Weibull model must provide four types of input:

(1) Cost model specifications
(2) The number of cost inputs
(3) An arbitrary 6-digit integer
(4) Cost data

These four types of input will be considered in turn.

The user must supply the cost model to the first computer program (WEIBTP) in the position shown in Fig. A1. This cost model is subject to several restrictions. First, it must be specified in FORTRAN IV.[7] Second, the total cost must be called output (IHIST). Third, the cost variables must be called FDATA(I) where I is any integral value from 1 up to the number of cost inputs that the user specifies.

An example of a user-specified cost model is shown in Fig. A2.

The second and third types of input must be provided on a punched card immediately following the first program, WEIBTP. The format of that card is:

Columns 2 to 4: An integer signifying the number of cost inputs. This number may take any integral value from 1 to 100

Columns 5 to 10: An arbitrary 6-digit number used to generate uniform random numbers

Columns 11 to 80: Blank

The final type of input provided by the user is the cost data. For each cost input the user must provide the most likely value, a low value and the probability that the actual value will be lower, and a high value and the probability that the actual value will be lower. Each value and probability is written on a punch card with the following type of format:

±X.YYE±ZZ

where ±X.YY raised to the power +ZZ is the value or probability. For example, −2700 would be written as −2.70E+03. The card format for each cost input is:

Columns 2 to 10: Low value

Columns 12 to 20: High value

Columns 22 to 30: Most likely value

Columns 32 to 40: Probability that actual value will be higher than the high value

Columns 42 to 50: Probability that actual value will be lower than the low value

Each cost data card is placed in a sequence such that its position in the stack of cost data cards is equal to the subscript of the cost input variable it represents. Thus the card containing FDATA(5) would be the fifth in the stack of cost data cards and that containing FDATA(15) would be fifteenth.

```
  888 RANDOM=FLOAT(L)/100000.
      IRANDM=L
      RANLOG=-ALOG(RANDOM)
  121 FDATA(IKE)=WYBARY(1,IKE)*RANLOG**(1./WYBARY(2,IKE))+WYBARY(3,IKE)
C
C   THE COST MODEL IS INSERTED HERE
C
C   *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *
C   *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *
      OUTPUT(IHIST)=FDATA(1)+FDATA(2)+FDATA(3)+FDATA(4)+FDATA(5)+FDATA
     *(6)
C
      IF (IHIST.GE.ITER) GO TO 9960
       IHIST = IHIST+1
       GO TO 9908
 9960 CONTINUE
```

Fig. A1—Position of the Cost Model in the Weibull Program, Example

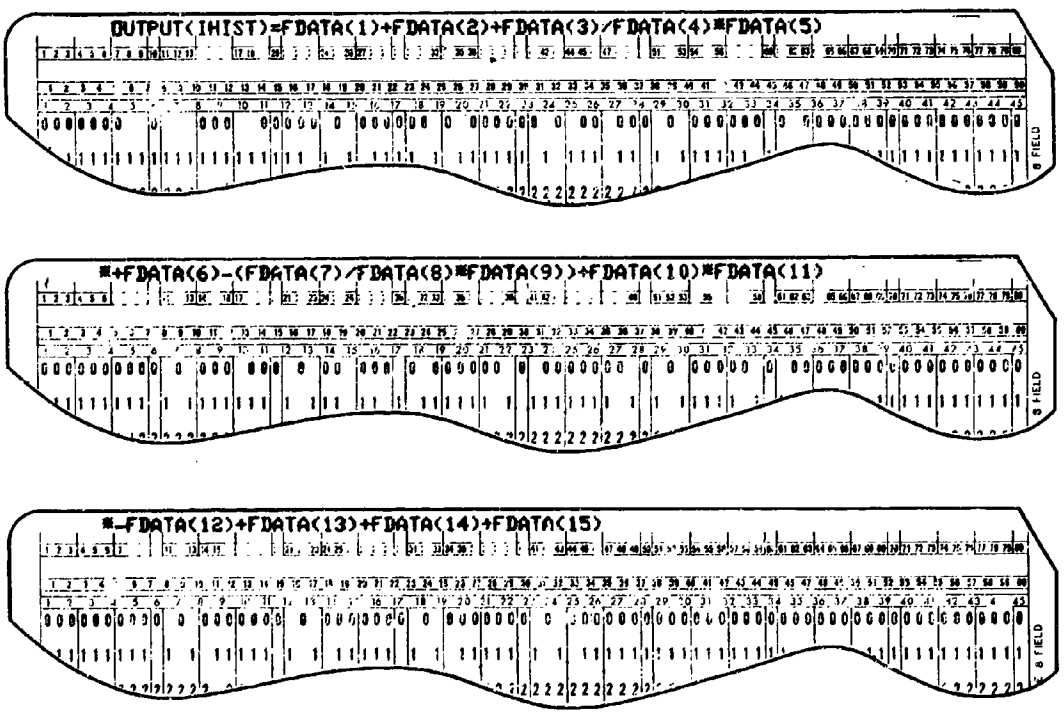


Fig. A2—User-Specified Cost Model, Example

RAC

To further elucidate the input requirements the following example is given. At a very high level of aggregation the total recurring costs of an infantry division can be thought of as the sum of operating costs, replacement costs, and pay.

For a planned infantry division the most likely annual operating cost might be $1.0 million, but there is a 10 percent chance that operating costs might be higher than $2.0 million or lower than $0.8 million. Similarly, annual replacement costs most likely would be $0.5 million with a 10 percent chance of being lower than $0.4 million or higher than $0.8 million. Annual pay most likely would be $0.5 million and would have a 25 percent chance of being $0.4 million and a 20 percent chance of being $0.6 million.

The user-supplied cost model is shown in the equation

```
OUTPUT(IHIST)=FDATA(1)+FDATA(2)+FDATA(3)
```

OUTPUT(IHIST) is total recurring cost, FDATA(1) is assumed to be operating cost, FDATA(2) is replacement cost, and FDATA(3) is military pay.

Next the user must supply a data card containing the number of cost inputs and a 6-digit random number. An example of such a card is shown in Fig. A3.

The 3 in col 4 signifies that there are three cost inputs (operating costs, replacement costs, and military pay). The number 987654 is an arbitrary integer used to generate random numbers.

Finally, the user must supply the input cost information. Since FDATA(1) refers to operating costs, the first input cost data card deals with operating cost. Fig. A4 shows the required input cost data card.

Note that the low, high, and most probable values are written in the form:

$$0.8 \times 10^6$$
$$2.0 \times 10^6$$
$$1.0 \times 10^6$$

The input cost data cards for replacement costs and pay are similarly prepared. The user-provided data (excluding the cost model) for this example are shown in Fig. A5.

To place the inputs in proper perspective to the rest of the computer program, Fig. A6 shows the program deck configuration. The user-provided inputs are indicated by arrows and the 7044 control cards are noted by asterisks.

BETA INPUT REQUIREMENTS

The Beta model, like its Weibull counterpart, requires four types of input:

(1) Cost model specifications
(2) The number of cost inputs
(3) An arbitrary 6-digit integer
(4) Cost data

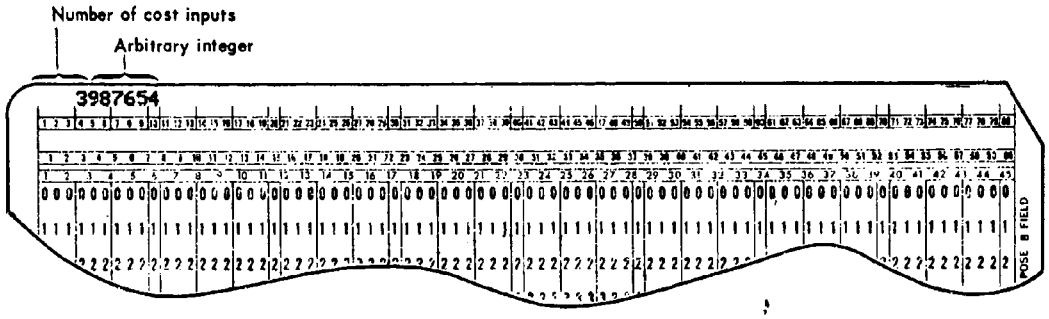


Fig. A3—Initial Data Card, Weibull Program

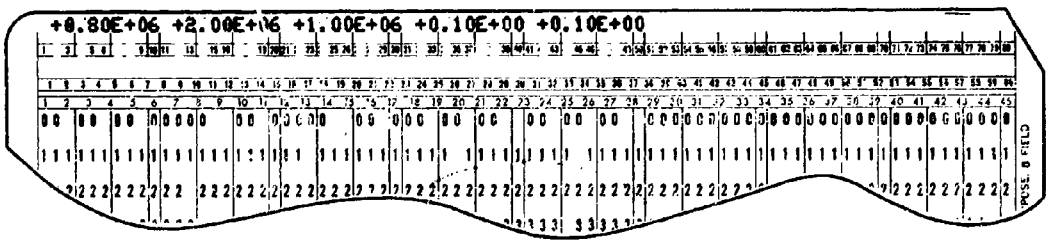


Fig. A4—Weibull Input Data, Example

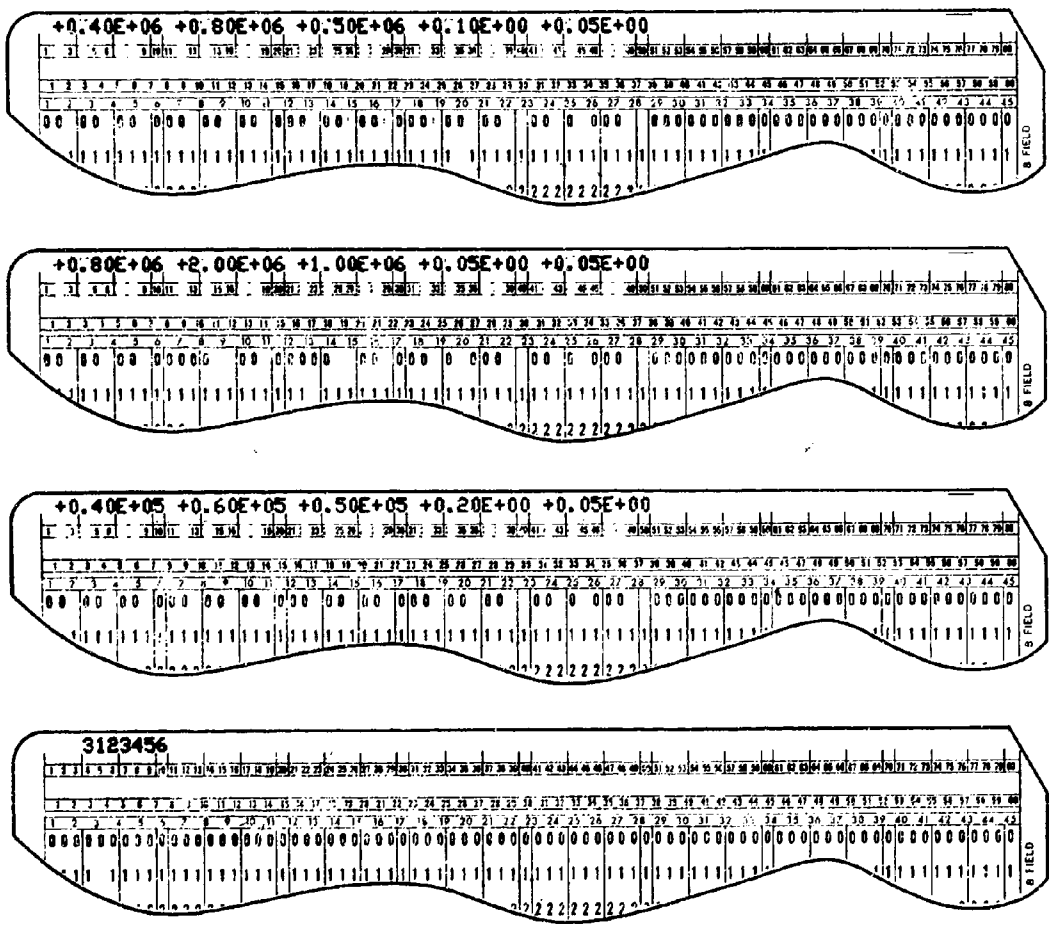


Fig. A5—User-Provided Data, Weibull Program

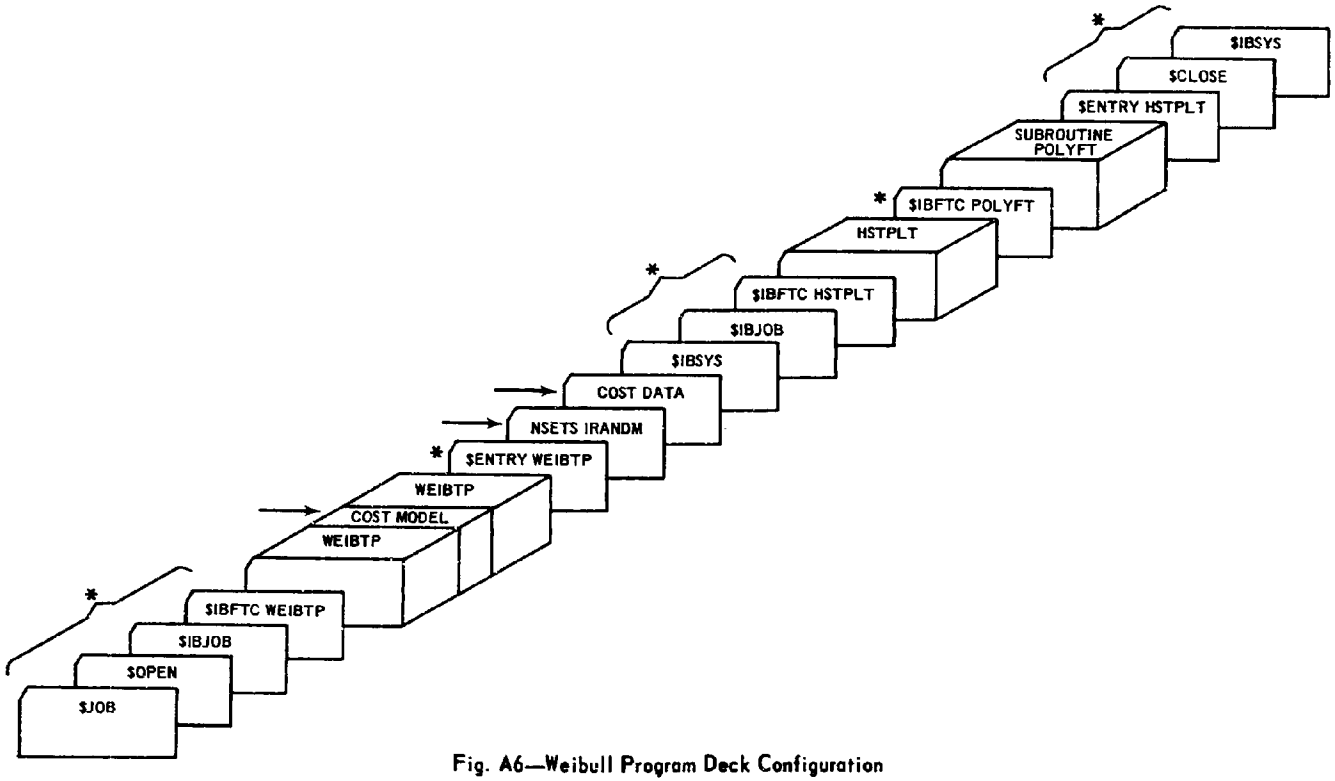


Fig. A6—Weibull Program Deck Configuration

* 7044 control cards ⟶ User-provided inputs

The first three types of input are identical to those required for the Weibull model. The cost data requirements are similar but not identical.

The user must imbed within the first program deck (called BETATP) a cost model. Figure A7 shows the position of the model within the program.

The cost model is subject to a number of restrictions. First, it must be written in FORTRAN IV.[7] Second, the total cost must be called output (IHIST). Third, the cost inputs must be called FDATA(I) where I is any integral value from 1 up to the number of cost inputs that the user specifies.

An example of a user-specified cost model is shown in Fig. A8.

The second and third types of user-supplied inputs must be placed on a punched card immediately following the third subroutine, EVLINT. The format of that card is:

Columns 2 to 4: An integer signifying the number of cost inputs. This number must be a positive integer no greater than 100.

Columns 5 to 10: An arbitrary 6-digit number used to generate uniform random numbers.

Columns 11 to 80: Blank.

The final type of input to be furnished by the user is cost data.

For each cost input the user must provide the most likely value, a low value, and a high value. Additionally, the user must choose a standard Beta distribution that best matches his conception of how the probability distribution would look. For each cost input the above cost data are written on a punched card with the following type of exponential notation:

$$\pm X.YYE \pm ZZ$$

which is equivalent to $\pm X.YY^{\pm ZZ}$. For example, +34000 would be written as +3.40E+04. The distribution type is not written using the above exponential notation but is represented as an integer.

The punched card format for each cost input is:

Columns 34 to 42: Low value

Columns 44 to 52: High value

Columns 54 to 62: Most likely value

Column 63: Distribution type. This type is represented by an integer with possible values ranging from 1 to 9. Figure A9 shows the type of distribution corresponding to each integer.

The cost input punched cards are placed immediately after the punched card containing the number of cost inputs and the arbitrary integer.

Each cost input card is placed in a sequence corresponding to the subscript of the cost input variable it represents. For example, the cost input card containing the costs associated with FDATA(5) would be fifth in the stack of cost input cards.

To illustrate the user-supplied inputs assume, for example, that the total recurring cost of an infantry division is the sum of operating cost, replacement cost, and pay. The most likely annual operating cost of a planned infantry division might be $1.0 million and the upper and lower limits are $2.0 million and $0.8 million respectively. Since it is more probable that operating cost, if not $1.0 million, will be higher and since operating cost is quite volatile,

```
   70  CONTINUE
       IHIST=1
 9908  CALL SAMPLE
C
C    SAMPLE RETURNS NSETS VALUES OF FDATA FOR USE AS INPUT TO THE COST MODEL
C
C    THE COST MODEL IS INSERTED HERE
       OUTPUT(IHIST)=FDATA(1)+FDATA(2)+FDATA(3)+FDATA(4)+FDATA(5)+
      *FDATA(6)+FDATA(7)
C
C    *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *
C    *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *
C
       IF (IHIST.GE.ITER) GO TO 9960
        IHIST = IHIST+1
        GO TO 9908
 9960 CONTINUE
```

Fig. A7—Position of the Cost Model in the Beta Program, Example

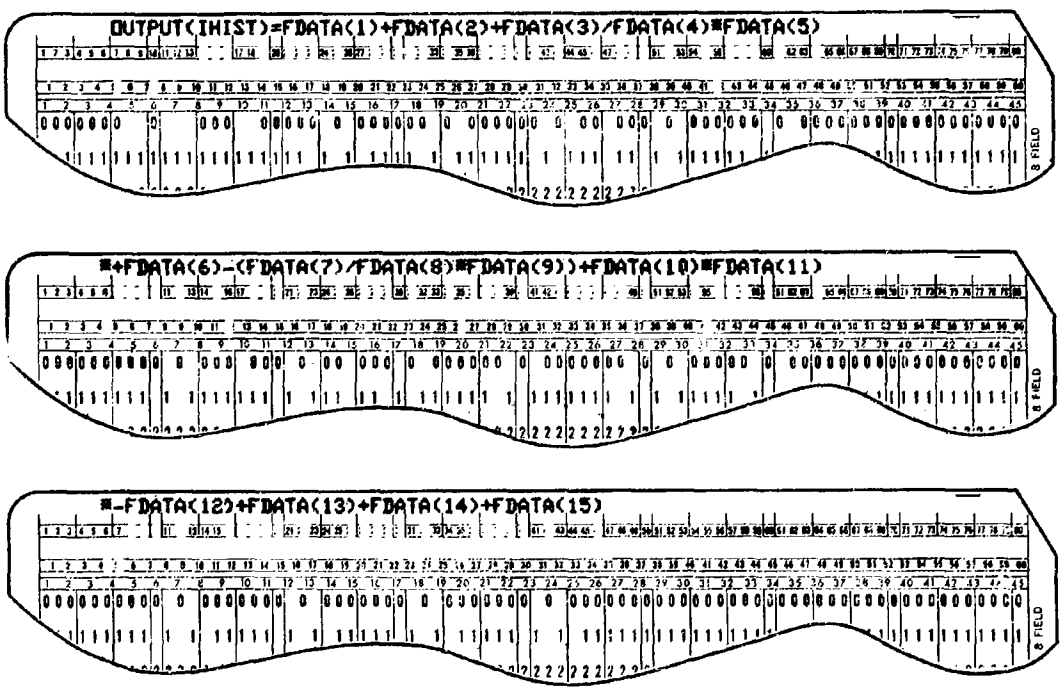


Fig. A8—User-Specified Cost Model, Example

the cost analyst might choose distribution type 3. Similarly, the annual replacement cost most likely, low, and high values are $0.5 million, $0.4 million, and $0.8 million. Like operating costs, replacement cost is quite volatile and if the most likely value is wrong it is probably too low. Therefore the cost analyst might choose type 3 once again. Finally, the most likely, low, and high values for military pay are $0.5 million, $0.4 million, and $0.6 million. Military

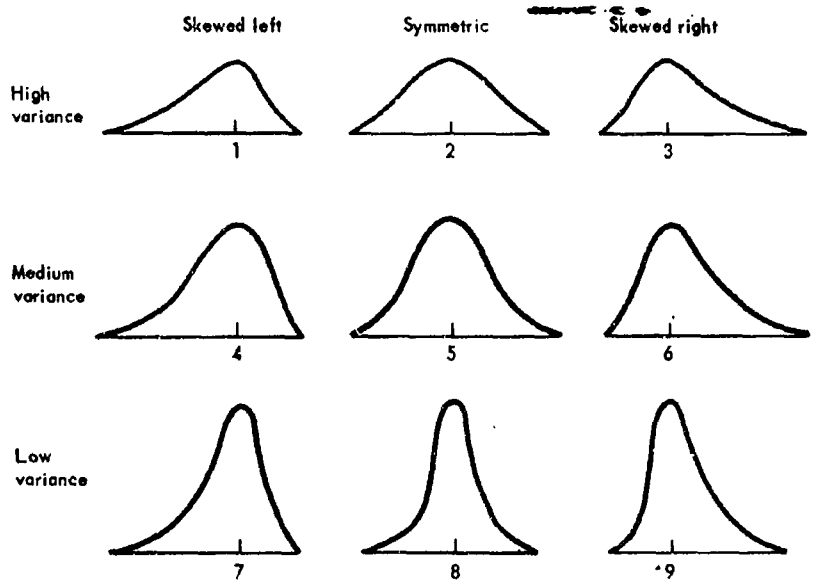


Fig. A9—Distribution Types

pay does not change much and the most likely estimate is equally likely to be too high or too low. Thus the cost analyst might consider distribution type 8 to be the closest approximation to his concept of the military pay distribution.

To run the above data in the Beta model, the user must first supply a cost model. The required cost model is

```
OUTPUT(IHIST)=FDATA(1)+FDATA(2)+FDATA(3)
```

OUTPUT(IHIST) is the total recurring cost, FDATA(1) is operating cost, FDATA(2) is replacement cost, and FDATA(3) is military pay.

Next a punched card containing the number of cost inputs and a 6-digit random number must be supplied. Such a card is shown in Fig. A10.

The 3 in col 4 of that figure specifies to the model that there are three cost inputs (operating cost, replacement cost, and military pay). The number 123456 is an arbitrary number used to generate random numbers.

Finally, the user must supply the input cost information. FDATA(1) refers to the first variable in the cost model. Examples of user-supplied input cost data cards are shown in Fig. A11.

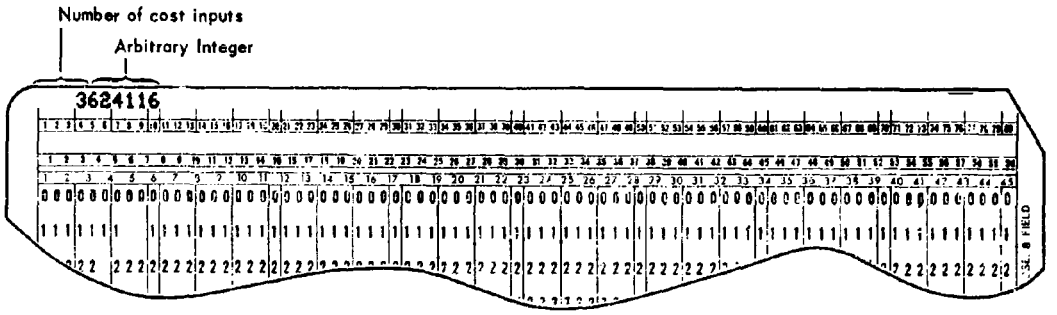


Fig. A10—Initial Data Card, Beta Program

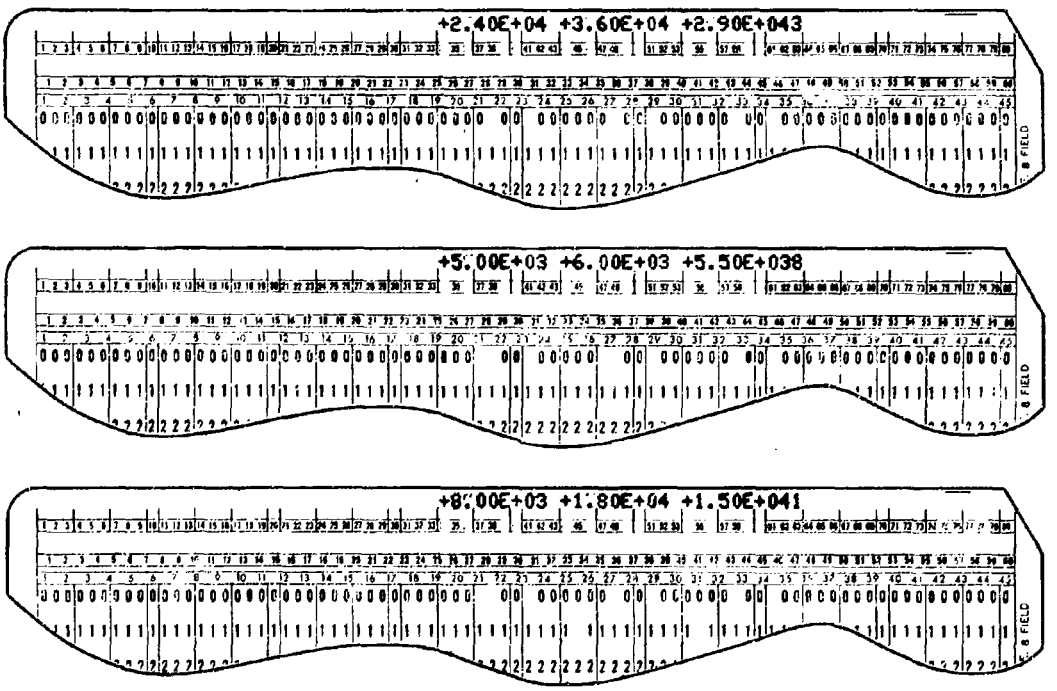


Fig. A11—Input Cost Data, Example

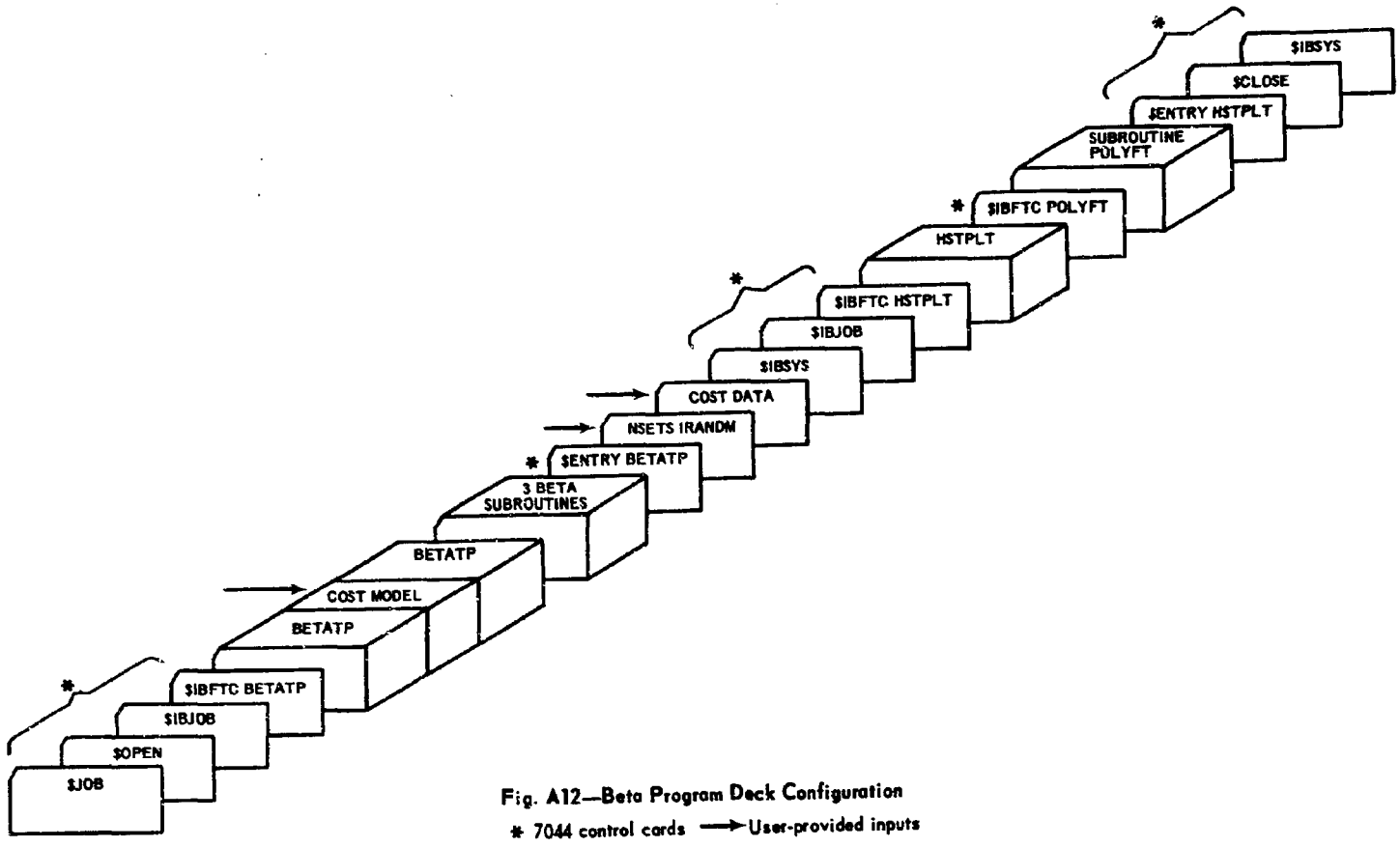


Fig. A12—Beta Program Deck Configuration

* 7044 control cards ⟶ User-provided inputs

To place the inputs in proper perspective to the rest of the computer program, Fig. A12 shows the program deck configuration. The user-provided inputs are indicated by arrows and the 7044 control cards are noted by asterisks.

ERROR MESSAGES

The error messages for the Weibull and the Beta programs are identical except for one message dealing with input distribution types for the Beta program. This section lists the error messages (data errors), explains the meaning, and suggests methods for correcting the errors.

DATA ERROR—MODE NOT BETWEEN HIGH AND LOW IN DATA SET XXX

Meaning: There is an error in cost data card XXX. The most probable value for cost variable XXX is either equal to or higher than the high value specified or equal to or less than the low value specified. All data are read but no simulation is run and no output is produced.

Corrective action: Check to see that the data are punched, from left to right, in the following order: low, high, and most likely. A second source of error to check is the exponent of one or more of the three data items. Finally, check to see that the data are punched in the correct columns.

DATA ERROR—LOW GREATER THAN HIGH IN DATA SET XXX

Meaning: There is an error in cost data card XXX. The low value specified on the card is greater than or equal to the high value. All data are read but no simulation is run and no output is produced.

Corrective action: Same as for the previous error message.

DATA ERROR—DISTRIBUTION TYPE FOR DATA SET XXX IS ZERO OR NOT SPECIFIED

Meaning: This message is only generated by the Beta program. The type parameter on cost data card XXX is either not specified or is specified as zero. All data are read but no simulation is run and no output is produced.

Corrective action: Check cost data card for the missing parameter. The parameter may be punched in the wrong column.

ERROR MESSAGE—RANGE OF TOTAL COST ESTIMATES IS LESS THAN TEN PLOT DELETED

Meaning: The simulated total cost estimates have a range smaller than 10.0. In this case the X axis cannot be designed and the plot is deleted. Program execution continues.

Corrective action: The cost data cards may be altered by increasing all the exponents.

COMPUTER HARDWARE AND SOFTWARE REQUIREMENTS

This section examines the computer facilities required to run the Monte Carlo programs. Since many of the requirements are themselves a function of the kind of computer used, the following discussion should be considered only a guideline.

Hardware

Four hardware features are required for the Monte Carlo programs.

First, there must be some method of reading punched cards since the program and data inputs are on punched cards.

Second, there must be sufficient main (core) storage to contain the program. No precise storage requirements can be formulated because of variations among computers; however, it can be stated that the variable arrays employed in the programs require 3000 words and the programs themselves are at least as large. Ten thousand words of storage ought to be sufficient.

Third, there must be some auxiliary storage device to hold intermediate results. This device might be a tape drive, a disk, or a drum. The capacity of the storage device must exceed 2000 words.

Finally, there must be a printer to print the output. The printer must have a capacity of at least 120 characters per line.

Software

The programs are written entirely in 7040/7044 FORTRAN IV. They do not use either "Print" or "Punch" statements.

The programs are designed to run under the 7040/7044 operating system. Use with any other computer will entail the user's supplying a different set of computer control cards.

PROGRAM LOGIC

The purpose of this section is to provide sufficient information about the programs per se to allow them to be modified successfully. A verbal description of the Weibull program and subprograms is supplied as well as a description of how they are linked together.* Complete program listings and flow charts are provided for both programs. The verbal description is keyed to the program listing. References are made to specific lines of computer coding. Knowledge of the flow charts, on the other hand, is not specifically required to read the verbal description. However, the flow charts should prove quite useful as a general guide to the programmer.

Weibull Program

There are three arrays used in the Weibull program (called WEIBTP). The first, FDATA, is the array containing the values of the cost input data. The subscript 100 sets the maximum number of cost inputs that may be used

*The Beta program is similar in format and employs identical output routines.

in a cost model. If this number is to be increased then FDATA and the second subscript of WYBARY, a second array used to simulate the FDATA, must be altered in the dimension statement. The first subscript of WYBARY is fixed. The third array, output, is dimensioned according to the number of iterations of the simulations to be run. If the number of iterations is to be increased then the subscript of output must be increased.

There are six internal program parameters that are set immediately after the dimension statement. First, there are three parameters, I01, I02, and I03, that set the device numbers for the system input device, the system output device, and an intermediate storage device. The fourth parameter, ITER, is set to the number of iterations of the simulation to be run. If ITER is increased, then the subscript of the array output must be increased. (An array in the plot routine must also be increased.) The fifth parameter, NPAGES, refers to the number of pages each plot is to occupy. NI, the sixth parameter, sets the number of class intervals to be used in constructing a frequency distribution of the simulation output. If NI is changed, a number of arrays in the plotting routine (discussed below) must be increased also.

After all internal parameters are set, two external parameters are read: NSETS, the highest subscript of FDATA actually used by the cost model, and IRANDM, an arbitrary 6-digit integer used to generate random numbers.

After the above-read statement, the next block of coding (through statement number 122) deals with reading the cost input data and defining the Weibull distribution around each input. First, the cost data are read into the computer. Then a check is made to see that the high cost is greater than the low cost. If it is not, an error message is written and a flag is set in the parameter ITER by setting ITER equal to 1. A second check is made to see if the mode falls within the range of the high and low. If it does not, an error message is written and ITER is set to 1. After the above checks are made ITER is checked to see if ITER = 1. If it does, the rest of the data is read and checked but no simulation takes place. Data are written on device I03 signaling that errors did occur and the run is stopped. If no error occurred, then initial parameters for a Newton-Raphson iterative process for specifying the Weibull distribution for a given input variable are specified. The iterative process is contained in the coding starting at statement number 3100 and extending to statement number 3110. The last four statements before statement number 122 transfer the three parameters specifying the Weibull distribution to WYBARY. Then control returns to the second read statement and the process is repeated until all the data are read and their Weibull distributions are specified.

The next block of coding, through statement number 9960, is the simulation. Basically the coding consists of a double loop, the inner running from statement number 9980 to statement number 121, and the outer from statement number 9980 to 9960. The inner loop generates uniformly distributed random numbers and then distributes them according to the Weibull distribution associated with the input data. The process is repeated until all input data have values. In the outer loop the cost model is evaluated on the basis of the values for the input data. The above process is repeated ITER times, i.e., until there are ITER values for the total cost output.

The next block of coding, to statement number 9961, writes the number of iterations, the number of class intervals, the number of pages for each plot,

the number of data inputs, and the simulation results on intermediate storage device I03. The storage device is backspread so as to be in position to be read and the program terminates.

The final section of coding contains the error messages and format statements.

Plot and Frequency Distribution Program

The plot and frequency distribution program (called HSTPLT) is separate from the Weibull and Beta programs. In an effort to reduce core storage requirements, HSTPLT is loaded on top of the preceding Weibull or Beta programs with all preliminary results being saved on an intermediate storage device.

The internal organization of HSTPLT includes a main program to plot the results of the simulation and a subprogram to transform the simulation data into a smooth curve. The program and subprogram are considered in turn.

The program HSTPLT is conceptually divided into two parts. The first part sets up the output histogram and calculates the mean and standard deviation of the data. The second part plots the output data. In the first part there are four arrays. NFREQ is dimensioned as large as NI, the number of class intervals in the histogram. NFREQ is the frequency of output in each class interval. XVAL is dimensioned the same as NFREQ and is the midpoint value for each class interval. A third array, SMOOTH, contains the smoothed frequencies corresponding to each value of XVAL and is also dimensioned the same as NFREQ. The final array is output having a dimension of ITER, a parameter that is set in the Weibull and Beta programs.

There are five internally set parameters. The first three, I01, I02, and I03, are the numbers of the system input device, the system output device, and an intermediate storage device, respectively. The fourth parameter, KOR, is the degree of the polynomial used to smooth the simulation output. The final parameter, IHOPE, is set to zero initially. IHOPE counts the number of times HSTPLT is executed. When IHOPE is 1, the program is operating on the unsmoothed data. When IHOPE is 2, the program is dealing with smoothed data. In addition to the above parameters there is a data statement that establishes four graphic characters for the Y axis of the output plots.

After the internal parameters are set, the program reads internal parameters saved on storage device I03. These parameters are described in the section dealing with the Weibull and Beta program logic. If the parameter ITER is equal to 1, then there was an input data error and execution is halted. Otherwise the simulation output is read.

Statement number 445 begins the main loop of the program. IHOPE is incremented to show the number of times that the loop is being executed. The first block of coding, through statement number 91, is concerned with finding the maximum and minimum values produced by the simulation. Immediately following statement number 91 is a block of coding that calculates the length for the plot and histogram class intervals. Statement 90 computes the midpoint of each class interval. After statement number 90, through statement number 92, the frequency or the number of observations falling within each class interval is calculated. The method is basically to start from the lowest

class interval and to test whether the upper limit of the class interval is greater than or equal to the value of an observation. If so, the number of observations within the class interval is increased by 1. Frequency calculation occurs only during the first execution of the main program loop, since during the second execution smoothed frequencies have been supplied by the smoothing subroutine. The remainder of the coding for the first portion of HSTPLT, through statement number 3200, consists of a straightforward calculation of the arithmetic mean for all observations and of the standard deviation.

The plotting routine is a conceptually separate part of HSTPLT. However, technically it is only a continuation of the coding for HSTPLT. There are three arrays. First, LABELX contains the labels used for the X axis of the output plot. A dimension of 16 is given to LABELX, with LABELX(16) set to a high value, although there are only 15 labels printed owing to an idiosyncrasy in the coding that prints the X axis. LABELY contains the Y axis labels and LINE contains the graphic characters used to generate the Y axis.

The first two lines of coding in the plotting routine determine the number of printer lines per class interval. The number of lines is at least 1. These two lines in conjunction with the value of NI control the number of pages the output plot will fill. If NI is greater than 50, then the number of pages is NI/ 50 (rounding upward to the next highest integer).

The next block of coding, through statement number 450, constructs the scale for the X axis and fills the X axis labels with the proper values. The fourth line of coding (the line preceding statement number 400) checks to see if the range of the values for the simulation output is greater than 10. If it is not, control is transferred to an error message routine. The coding starting at statement number 400 and ending at statement number 420 constructs the increments for the X axis labels. The coding from statement numbers 420 to 432 establishes the value of the first label and the remaining coding, through statement 450, establishes the values for each label.

The next five lines of coding, through statement number 460, constructs the Y axis increments (first two lines) and determines the Y axis label values (last three lines).

The next section, through two lines past statement number 483, writes the Y axis labels, the Y axis itself, and the minimum value of X.

The next block of coding, extending through statement number 499, prints the X axis labels, the plot, and the mean value of X. The inner loop in this coding (DO 499 J=1, INTWID) is essentially unused unless the number of class intervals (NI) has been reset to a value less than 26. The plot logic will not be discussed in detail. However, the coding idiosyncrasy with respect to LABELX (16) noted above will be explained. After the last label has been printed, by statement number 471, the subscript of LABELX is increased to 16. If LABELX (16) is less than the minimum value of X, then the first "IF" statement after statement number 475 will eventually cause statement number 471 to be executed one extra time.

After completing the plot the maximum value of X and the standard deviation of X are written. If IHOPE is 2, i.e., if this is the second time through the main program, execution is halted. If this is the first time, the smoothing routine is called. On return from the smoothing routine, the smoothed values are rounded to the nearest integer and control is transferred to the beginning of HSTPLT.

The smoothing subroutine is SHARE subroutine FOLYFT and documentation is available through the SHARE organization. In general, the subroutine fits a polynomial to the unsmoothed data using the least-squares criterion.

The dimension of the arrays XVAL, NFREQ, and SMOOTH must equal NI, the number of class intervals. The dimension of C, SMYX, and AMEANX must equal KOR, the degree of the polynomial being used to smooth the data. The dimension of A must be KOR squared and the dimension of SUMX must be two times KOR.

A series of flow charts, Figs. A 13 to A19, follows, in which various routines of the Monte Carlo simulation are portrayed.

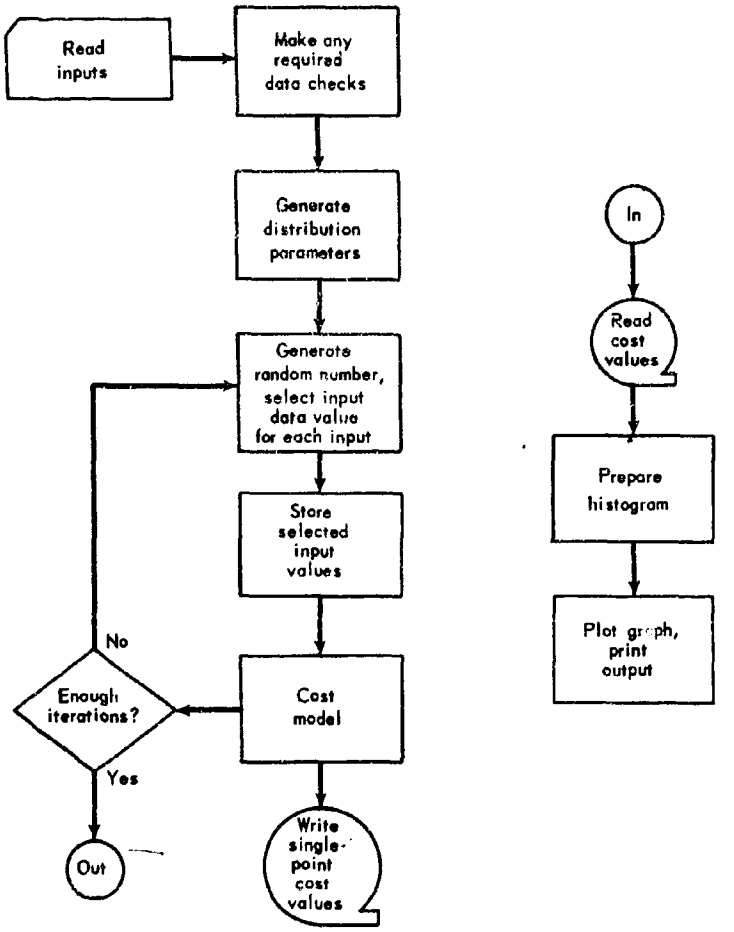


Fig. A13—General Flow Chart of Monte Carlo Routines

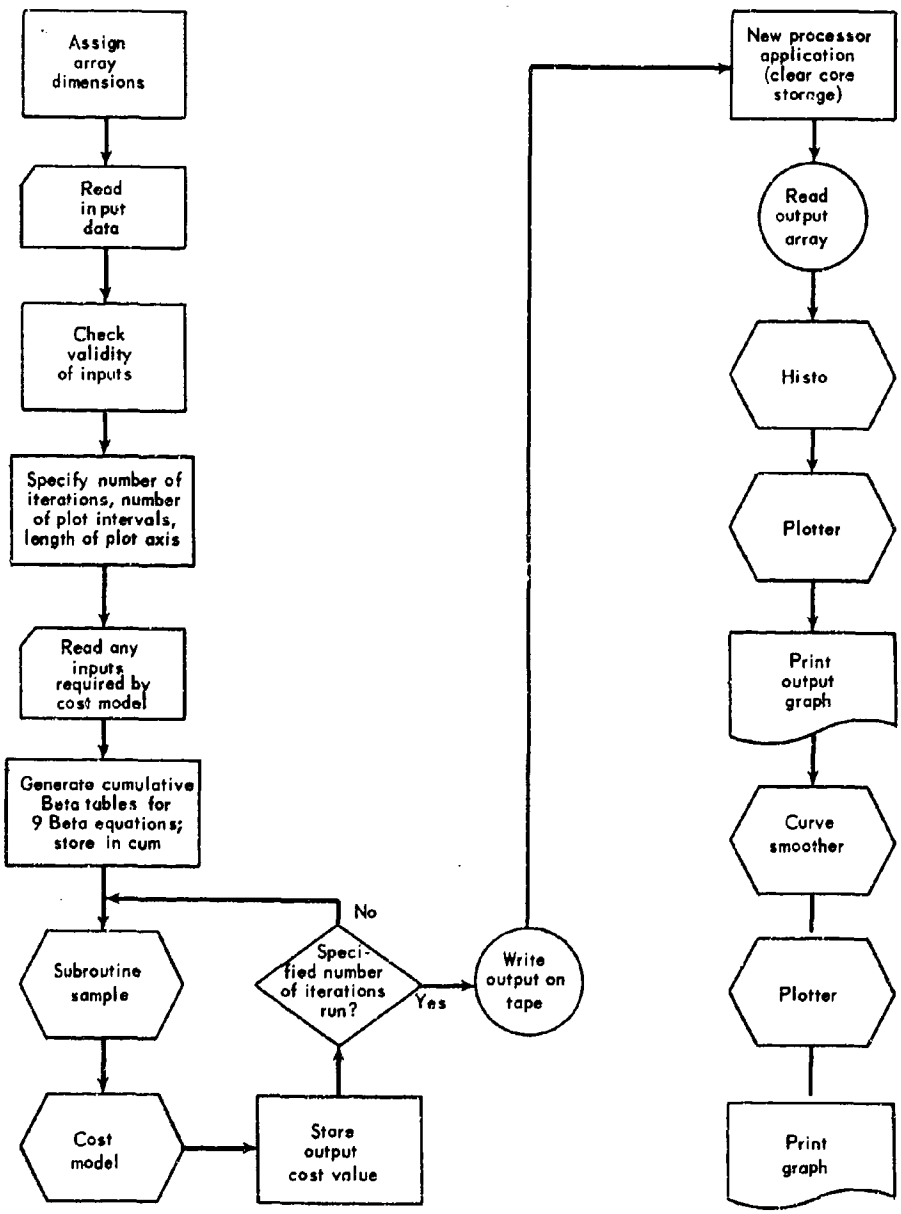


Fig. 14—Flow Chart of BETATP Cost Processing Routines

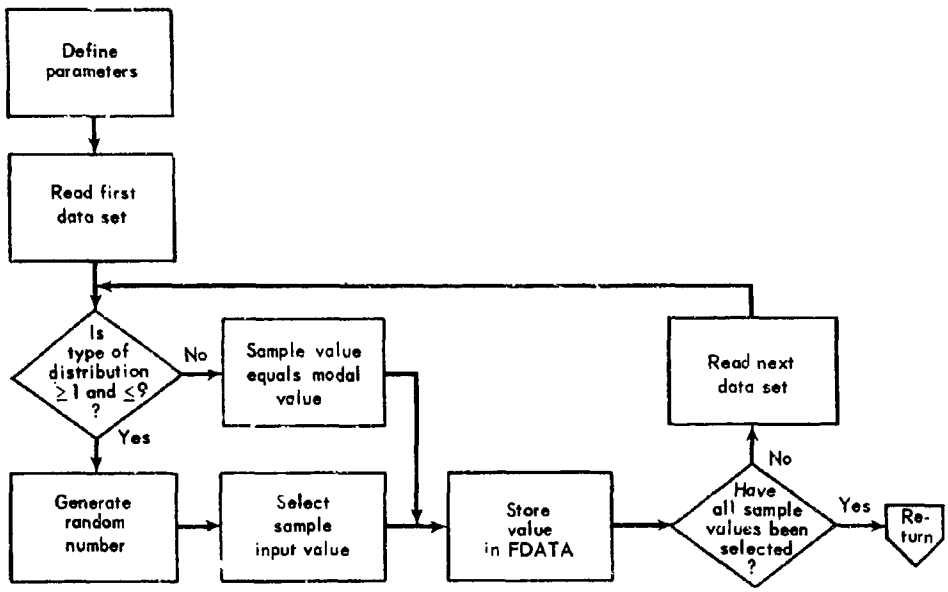


Fig. A15—Flow Chart of Sample Routine

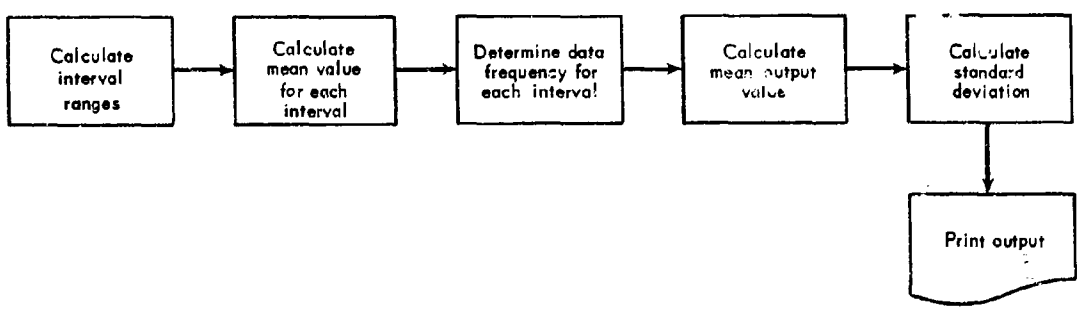


Fig. A16—Flow Chart of HISTU Routine

Fig. A17—Flow Chart of Plotting Routine

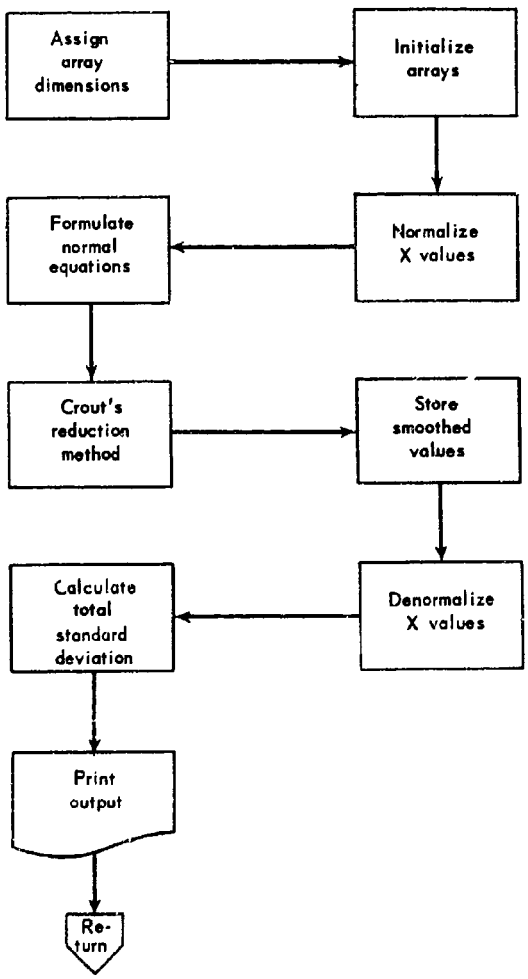


Fig. A18—Flow Chart of POLYFT Routine

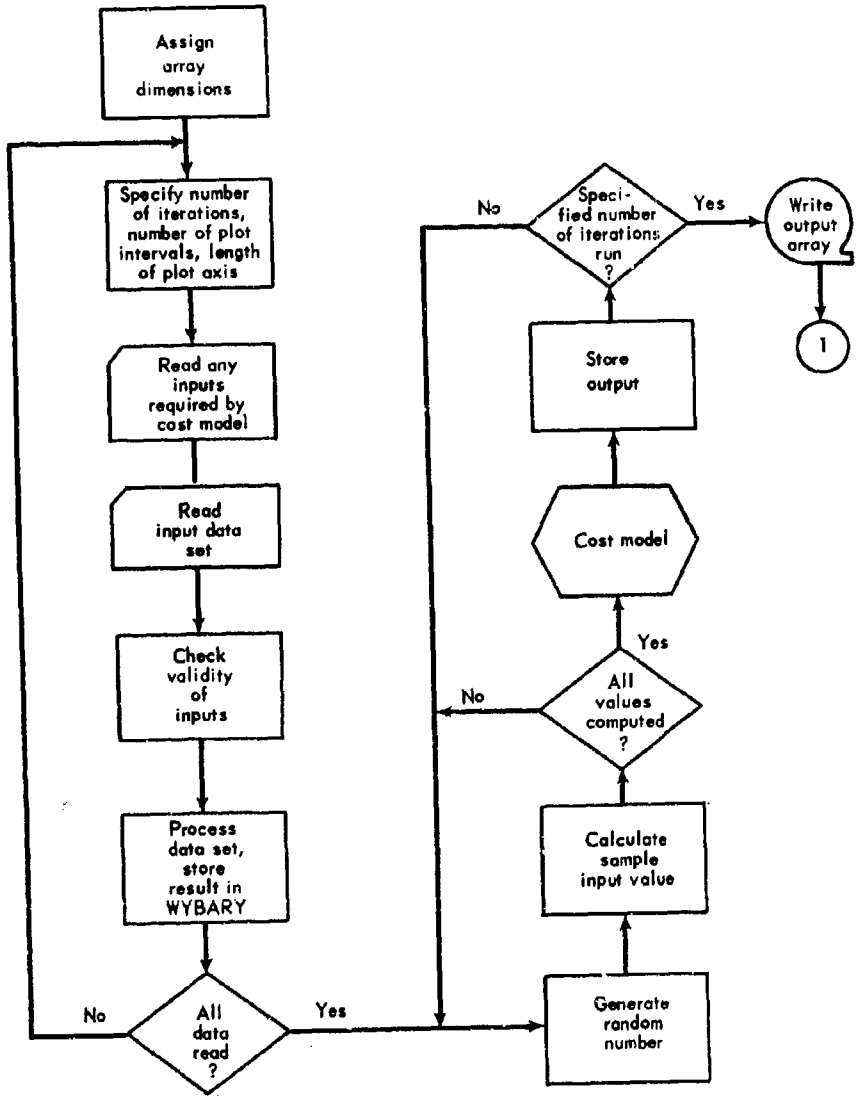


Fig. A19—Flow Chart of WEIBTP Routine

WEIBULL PROGRAM LISTING

```
$JOB            009.303,JOHNSON,WW,MONTE
$OPEN           S.SU10,REWIND
$IBJOB MCCOST
$IBFTC WEIBTP  NODECK
      DIMENSION FDATA(100),OUTPUT(1000),WYBARY(3,100)
      IO1=5
      IO2=6
      IO3=10
      ITER=1000
      NPAGES=2
      NI=100
      READ(IO1,103) NSETS,IRANDM
      DO 122 JOT=1,NSETS
      READ (IO1,104) V2,V1,V0,P1,P2
      IF ((V1-V2).LT.0.) GO TO 9961
 9963 IF (V0.LE.V2 .OR. V0.GE.V1) GO TO 9962
 9964 IF (ITER .EQ. 1 ) GO TO 122
      IF ((V0-V2).LT.(V1-V0)) GO TO 1
      EM1=4.0
      EM2=3.5
      GO TO 2
    1 EM1=3.0
      EM2=3.5
    2 P=1.-P1
      PLN=-ALOG(P)
      A1=(PLN*(EM1/(EM1-1.)))**(1./EM1)
      EK1=(V1-V0*A1)/(1.-A1)
      B1=-((EM1-1.)/EM1)*((V2-EK1)/(V0-EK1))**EM1
      P2A=EXP(B1)
      F1=P2-P2A
 3100 A2=(PLN*(EM2/(EM2-1.)))**(1./EM2)
      EK2=(V1-V0*A2)/(1.-A2)
      B2=-((EM2-1.)/EM2)*((V2-EK2)/(V0-EK2))**EM2
      IF (B2.GT.(-89.5)) GO TO 105
      P2B=0.
      GO TO 106
  105 P2B=EXP(B2)
  106 F2=P2-P2B
      IF(F2.LT..0001) GO TO 3110
      EK1=EK2
      P2A=P2B
      EMTEMP=EM2
      EM2=EM2-F2*((EM2-EM1)/(F2-F1))
      EM1=EMTEMP
      F1=F2
      GO TO 3100
 3110 FLAMDA=(V0-EK2)*(EM2/(EM2-1.))**(1./EM2)
      WYBARY(1,JOT)=FLAMDA
      WYBARY(2,JOT)=EM2
      WYBARY(3,JOT)=EK2
  122 CONTINUE
      IF (ITER .EQ. 1 ) GO TO 9960
      IHIST=1
      DATA JK/100043/
 9908 DO 121 IKE=1,NSETS
      L=IRANDM
  777 L=78125*L
      L=L-(L/JK)*JK
      IF(L-100000) 888,888,777
  888 RANDOM=FLOAT(L)/100000.
```

```
      IRANDM=L
      RANLOG=-ALOG(RANDOM)
  121 FDATA(IKE)=WYBARY(1,IKE)*RANLOG**(1./WYBARY(2,IKE))+WYBARY(3,IKE)
C
C     THE COST MODEL IS INSERTED HERE
C
C     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *
C     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *
      OUTPUT(IHIST)=FDATA(1)+FDATA(2)+FDATA(3)+FDATA(4)+FDATA(5)+FDATA
     *(6)
C
      IF (IHIST.GE.ITER) GO TO 9960
       IHIST = IHIST+1
       GO TO 9908
 9960 CONTINUE
      WRITE(IO3) ITER,NI,NPAGES,NSETS
      WRITE(IO3) (OUTPUT(K),K=1,ITER)
      BACKSPACE 10
      BACKSPACE 10
      CALL EXIT
 9961 WRITE(IO2,100) JOT
  100 FORMAT(55H DATA ERROR -- LOW GREATER THAN HIGH IN DATA SET NUMBER,
     *I3)
  107 FORMAT(56H DATA ERROR -- MODE NOT BETWEEN HIGH AND LOW IN DATA SET
     *,I4)
      ITER=1
      GO TO 9963
 9962 WRITE (IO2,107) JOT
      ITER=1
      GO TO 9964
  103 FORMAT (I4,I6)
  104 FORMAT (5E10.2)
       END
$ENTRY          WEIBTP
   6431763
 +1.30E+04 +2.00E+04 +1.54E+04 +0.01E+00 +0.01E+00
 +5.00E+03 +6.00E+03 +5.45E+03 +0.01E+00 +0.01E+00
 +1.80E+04 +2.50E+04 +2.10E+04 +0.01E+00 +0.01E+00
 +1.00E+04 +1.50E+04 +1.17E+04 +0.01E+00 +0.01E+00
 +1.50E+04 +2.00E+04 +1.76E+04 +0.01E+00 +0.01E+00
 +1.00E+04 +1.15E+04 +1.06E+04 +0.01E+00 +0.01E+00
$IBSYS
$IBJOB COMPLT
$IBFTC HSTPLT  NODECK
C     EXTRACT OF HISTO
      DIMENSION NFREQ(100),XVAL(100),SMOOTH(100),OUTPUT(1000)
      INTEGER BLANK,X,DOT,APOS
      DATA BLANK,X,DOT,APOS/1H ,1HX,1H.,1H'/
      IO1=5
      IO2=6
      IO3=10
      READ (IO3) ITER,NI,NPAGES,NSETS
      IF (ITER .EQ.1 ) STOP
      READ(IO3) (OUTPUT(K),K=1,ITER)
      ITER25=ITER/25
      KOR=25
C     CALCULATES MEAN, STANDARD DEVIATION, AND PREPARES HISTOGRAM
      IHOPE=0
  445 IHOPE=IHOPE+1
C     CALCULATE INTERVAL RANGE (RINT) FOR NI INTERVALS
      XMIN = OUTPUT(1)
```

```
      XMAX = OUTPUT(1)
       DO 91 K = 1,ITER
      IF(OUTPUT(K).LT.XMIN) XMIN=OUTPUT(K)
      IF(OUTPUT(K).GT.XMAX) XMAX=OUTPUT(K)
   91 CONTINUE
      RINT = (XMAX-XMIN)/FLOAT(NI)
C     CALCULATE MEAN VALUE (XVAL) FOR EACH INTERVAL
      XVAL(1)=XMIN + RINT/2.
       DO 90 N = 2,NI
   90 XVAL(N)=XVAL(N-1) + RINT
C     DETERMINE DATA FREQUENCY FOR EACH INTERVAL
      IF(IHOPE.EQ.2) GO TO 93
      DO 20 N =1,NI
   20 NFREQ(N)=0
       DO 92  K = 1,ITER
      DO 21  N = 1,NI
      XLIM=XVAL(N)+RINT/2.
      IF(XLIM.GE.OUTPUT(K)) GO TO 92
   21 CONTINUE
      N = NI
   92 NFREQ(N)=NFREQ(N)+1
C     CALCULATE MEAN VALUE (XMEAN)
   93 SUMX = 0.
      DO 30  K = 1,ITER
   30 SUMX=SUMX+OUTPUT(K)
      XMEAN = SUMX/FLOAT(ITER)
C     CALCULATE STANDARD DEVIATION (XSDEV)
      SUMSQ = 0.
      DO 40  K = 1,ITER
   40 SUMSQ=SUMSQ+(OUTPUT(K)-XMEAN)**2
      IF (ITER.NE.1) GO TO 3210
      XSDEV=0.
      GO TO 3200
 3210 XVAR = SUMSQ/(FLOAT(ITER)-1.)
      XSDEV = SQRT(XVAR)
 3200 CONTINUE
C
C     PLOTTING ROUTINE
C
      DIMENSION LABELX(16),LABELY(10),LINE(132)
      LABELX(16)=999999
C
C     DETERMINES WIDTH OF PLOTTING INTERVAL
C
      INTWID=50/NI
      IF(INTWID.EQ.0) INTWID=1
C
C     DESIGNS X AXIS SCALE
C
      XRANGE=XMAX-XMIN
      IFACTR=0
      XINCR=XRANGE/10.
      IF(XINCR.LT.1.) GO TO 498
  400 IF(XINCR.GE.1..AND.XINCR.LE.10.) GO TO 410
      XINCR=XINCR/10.
      IFACTR=IFACTR+1
      GO TO 400
  410 INCRX=XINCR
      IF(INCRX.EQ.3) INCRX=2
      IF(INCRX.GT.3.AND.INCRX.LE.7) INCRX=5
```

```
      IF(INCRX.EQ.8.OR.INCRX.EQ.9) INCRX=10
      IF(IFACTR.EQ.0) GO TO 420
      DO 420 I=1,IFACTR
      INCRX=10*INCRX
  420 CONTINUE
      IFACTR=0
      SCALOW=XMIN
  425 IF(ABS(SCALOW).GE.0. .AND. ABS(SCALOW).LE.10.) GO TO 430
      SCALOW=SCALOW/10.
      IFACTR=IFACTR+1
      GO TO 425
  430 LOSCAL=SCALOW
      IF(IFACTR.EQ.0) GO TO 440
      DO 440 I=1,IFACTR
      LOSCAL=10*LOSCAL
  440 CONTINUE
  431 IF(FLOAT(LOSCAL+INCRX) .GE. XMIN) GO TO 432
      LOSCAL=LOSCAL+INCRX
      GO TO 431
  432 CONTINUE
      LABELX(1)=LOSCAL+INCRX
      DO 450 I=2,15
  450 LABELX(I)=LABELX(I-1)+INCRX
C
C     LABELX NOW CONTAINS 15 INCREMENTAL VALUES TO BE USED
C        FOR SCALE MARKINGS ALONG THE X AXIS
C
C     DESIGN Y AXIS SCALE
C
      MAXY=(ITER*5)/NI
      INCRY=MAXY/10
      LABELY(1)=INCRY
      DO 460 I=2,10
  460 LABELY(I)=LABELY(I-1)+INCRY
C
C     LABELY NOW CONTAINS 10 INCREMENTAL VALUES TO BE USED
C         FOR SCALE MARKINGS ALONG THE Y AXIS
C
C
C     PRINT Y SCALE
C
      WRITE (IO2,402)(LABELY(I),I=1,10)
      DO 469 I=16,116,10
  469 LINE(I)=DOT
      DO 481 I=1,15
  481 LINE(I)=BLANK
      DO 482 I=17,115,2
  482 LINE(I)=BLANK
      DO 483 ISTART=18,108,10
      ISTOP=ISTART+6
      DO 483 I=ISTART,ISTOP,2
  483 LINE(I)=APOS
      WRITE(IO2,403)(LINE(I),I=1,116)
      WRITE(IO2,404)XMIN
C
C     BEGIN PLOTTING DATA POINTS AND LABELING X AXIS
C
      DX = XRANGE/ (FLOAT(INTWID)*FLOAT(NI))
      DY = FLOAT(MAXY)/100.
      XVALU=XMIN
      LAB=1
```

```
      DO 499 K=1,NI
      DO 499 J=1,INTWID
      YVALU=0.
      I=1
  477 IF (FLOAT(NFREQ(K)).LE.YVALU) GO TO 475
      LINE(I)=BLANK
      I=I+1
      YVALU=YVALU+DY
      GO TO 477
  475 LINE(I)=X
      XVALU=XVALU+DX
C
C     PRINT X VALUE IF APPLICABLE
C
      IF(FLOAT(LABELX(LAB)) .GT. XVALU) GO TO 470
C
C     PRINT XMEAN IF IT OCCURS IN THIS INTERVAL
C
      IF(XMEAN.LT.XVALU .OR. XMEAN.GE.(XVALU+DX)) GO TO 471
      GO TO 472
  470 IF (XMEAN.LT.XVALU .OR. XMEAN.GE.(XVALU+DX)) GO TO 473
      GO TO 474
  471 WRITE(IO2,405)LABELX(LAB),(LINE(II),II=1,I)
      LAB=LAB+1
      GO TO 499
  472 WRITE(IO2,406)LABELX(LAB),XMEAN,(LINE(II),II=12,I)
      LAB=LAB+1
      GO TO 499
  473 WRITE(IO2,407)(LINE(II),II=1,I)
      GO TO 499
  474 WRITE(IO2,408)XMEAN,(LINE(II),II=12,I)
  499 CONTINUE
      WRITE(IO2,411)XMAX
      WRITE(IO2,412)XSDEV
  412 FORMAT (///,9H STD DEV=,F10.3)
      GO TO 99
  498 WRITE(IO2,409)
   99 CONTINUE
      IF(IHOPE.NE.1) CALL EXIT
      CALL POLYFT (XVAL,NFREQ,NI,KOR,SMOOTH)
      DO 505 I=1,NI
  505 NFREQ(I)=SMOOTH(I) + .500
      GO TO 445
  402 FORMAT(1H1,12X,4H 000,10(6X,I4))
  403 FORMAT(116A1/)
  404 FORMAT(6HXMIN =,F10.3//)
  405 FORMAT(I6,9X,100A1)
  406 FORMAT(I6,3X,7HXMEAN =,F10.3,88A1)
  407 FORMAT(15X,100A1)
  408 FORMAT(9X,7HXMEAN =,F10.3,88A1)
  409 FORMAT (79HOERROR MESSAGE --- RANGE OF TOTAL COST ESTIMATES IS LES
     *S THAN TEN. PLOT DELETED )
  411 FORMAT(6HXMAX =,F10.3)
      END
$IBFTC FTKURV
      SUBROUTINE POLYFT(XVAL,NFREQ,NI,KOR,SMOOTH)
      IO1=5
      IO2=6
      IO3=10
C                                                          POLY0010
```

```
C       DIMENSION  FOR ARGUMENTS                                    POLY0020
        DIMENSION XVAL(100),NFREQ(100),A(625),C(25),SMOOTH(100)
C                                                                   POLY0040
C       DIMENSION  FOR SELF-GENERATED VALUES                        POLY0050
        DIMENSION  SUMX(100),SMYX( 50), AMEANX( 50)                 POLY0060
        PTS=FLOAT(NI)
        KTOR= 2*KOR                                                 POLY0090
C                                                                   POLY0100
C       INITIALIZATION                                              POLY0110
        DO  1  I=1,KOR                                              POLY0120
        SMYX(I)= 0.0                                                POLY0130
    1   AMEANX(I)= 0.0                                              POLY0140
        DO  2  I=1,KTOR                                             POLY0150
    2   SUMX(I)= 0.0                                                POLY0160
        SUMY= 0.0                                                   POLY0170
C                                                                   POLY0180
C       NORMALIZATION WITH RESPECT TO XMAX                          POLY0190
        XVAL1=XVAL(1)
        DO 707 NOJ=1,NI
  707   XVAL(NOJ)=XVAL(NOJ)-XVAL1
        XMAX=XVAL(NI)
        DO 102 I=1,NI
  102   XVAL(I)=XVAL(I)/XMAX
C                                                                   POLY0270
C                                                                   POLY0280
C       FORMULATION OF NORMAL EQUATIONS                             POLY0290
        DO  3 J=1,KTOR                                              POLY0300
        DO 3 I=1,NI
    3   SUMX(J)=SUMX(J)+XVAL(I)**J
        DO 4 I=1,NI
    4   SUMY=SUMY+FLOAT(NFREQ(I))
        AMEANY= SUMY/PTS                                            POLY0350
        DO  6 J=1,KOR                                               POLY0360
        AMEANX(J)= SUMX (J)/PTS                                     POLY0370
        DO 6 I=1,NI
    6   SMYX(J)=SMYX(J)+FLOAT(NFREQ(I))*XVAL(I)**J
        DO  8 I=1,KOR                                               POLY0400
        C(I)= SMYX(I) -PTS*AMEANX(I)*AMEANY                         POLY0410
        DO  8 J=1,KOR                                               POLY0420
        K= I+J                                                      POLY0430
        IJ =(J-1)*KOR +I                                            POLY0440
    8   A(IJ) =SUMX(K) -PTS* AMEANX(I)*AMEANX(J)                    POLY0450
C                                                                   POLY0460
C       CROUT'S  REDUCTION  METHOD                                  POLY0470
        DO  11 I=2,KOR                                              POLY0480
        III=(I-1)*KOR + 1                                           POLY0490
   11   A(III) = A(III)/ A(1)                                       POLY0500
        DO  12 J=2,KOR                                              POLY0510
        KM= J-1                                                     POLY0520
        DO 14 I=J,KOR                                               POLY0530
        AP1= 0.0                                                    POLY0540
        DO114 K=1,KM                                                POLY0550
        IK =(K-1)* KOR+ I                                           POLY0560
        KJ =(J-1)* KOR+ K                                           POLY0570
  114   AP1 = AP1 + A(IK) *A(KJ)                                    POLY0580
        IJ =(J-1)* KOR+ I                                           POLY0590
   14   A(IJ) = A(IJ) -AP1                                          POLY0600
        JP= J+1                                                     POLY0610
```

```
      IF (JP- KOR) 444, 444, 445                                        POLY0620
  444 DO 16 I=JP,KOR                                                    POLY0630
      AP1= 0.0                                                          POLY0640
      DO116 K=1,KM                                                      POLY0650
      JK =(K-1)* KOR + J                                                POLY0660
      KI =(I-1)* KOR + K                                                POLY0670
  116 AP1 =AP1 +A(JK) *A(KI)                                            POLY0680
      JI =(I-1)* KOR + J                                                POLY0690
      JJ =(J-1)* KOR + J                                                POLY0700
   16 A(JI) = (A(JI) -AP1)/A(JJ)                                        POLY0710
  445 DUMMY= 0.0                                                        POLY0720
   12 CONTINUE                                                          POLY0730
      C(1) =C(1)/A(1)                                                   POLY0740
      DO  18 I=2,KOR                                                    POLY0750
      AP1= 0.0                                                          POLY0760
      IM=I-1                                                            POLY0770
      DO 118 K=1,IM                                                     POLY0780
      IK =(K-1)* KOR + I                                                POLY0790
  118 AP1=AP1 +A(IK) *C(K)                                              POLY0800
      II =(I-1)* KOR + I                                                POLY0810
   18 C(I) =(C(I)- AP1) / A(II)                                         POLY0820
      KORM= KOR-1                                                       POLY0830
      IF (KORM) 122, 123,122                                            POLY0840
  122 DO  21 I=1,KORM                                                   POLY0850
      AP1= 0.0                                                          POLY0860
      M= KOR-I                                                          POLY0870
      MP= M+1                                                           POLY0880
      DO 121 K=MP,KOR                                                   POLY0890
      MK =(K-1)* KOR +M                                                 POLY0900
  121 AP1 =AP1 + A(MK)* C(K)                                            POLY0910
   21 C(M) =C(M) -AP1                                                   POLY0920
  123 AP1= 0.0                                                          POLY0930
      DO 24  I=1,KOR                                                    POLY0940
   24 AP1 =AP1 +AMEANX(I) *C(I)                                         POLY0950
      CO  = AMEANY -AP1                                                 POLY0960
C                                                                       POLY0970
  778 SRES = 0.0                                                        POLY1030
      WRITE(IO2,29)
   29 FORMAT (60H1          COST                FREQ      SMOOTHED
     * RES/43H       INTERVAL                            FREQ)
      DO 77 I=1,NI
      SMOOTH(I)=CO
      DO  27 J=1,KOR                                                    POLY1090
   27 SMOOTH(I)=SMOOTH(I)+C(J)*XVAL(I)**J
      RES=FLOAT(NFREQ(I))-SMOOTH(I)
C                                                                       POLY1130
C     DENORMALIZATION WITH RESPECT TO XMAX                              POLY1140
      XVAL(I)=XVAL(I)*XMAX
      XVAL(I)=XVAL(I)+XVAL1
   30 FORMAT( 4F15.4)                                                   POLY1160
      FREQ=FLOAT(NFREQ(I))
   77 WRITE(IO2,30)XVAL(I),FREQ,SMOOTH(I),RES
      RETURN                                                            POLY1270
      END                                                               POLY1280
$ENTRY          HSTPLT
$IBSYS
$CLOSE          S.SU10
```

BETA PROGRAM LISTING

```
$JOB            009,303,JOHNSON,WW,      MAIN        X7131
$OPEN           S,SU10,REWIND
$IBJOB MCCOST   NODECK
$IBFTC BETATP   NODECK
      DIMENSION OUTPUT(1000)
      COMMON/NONAME/NTABLE,NSETS,IRANDM,XTABLE(128),FDATA(100),
     * FMODE(100),FRANGE(100),NTYPE(100),ITER,NI,CUM(9,128)
      EXTERNAL SFX
      ITER=1000
      NI=100
      NPAGES=2
      IO1=5
      IO2=6
      IO3=10
      READ(IO1,102) NSETS,IRANDM
  102 FORMAT (I4,I6)
      DO 708 I=1,NSETS
      READ (IO1,103) FLOW,FHIGH,FMODE(I),NTYPE(I)
      FRANGE(I)=FHIGH-FLOW
      IF(FRANGE(I).LT.0.) GO TO 9961
  709 IF(NTYPE(I).GE.1.AND.NTYPE(I).LE.9) GO TO 707
      WRITE (IO2,101) I
      ITER=1
  707 IF (FMODE(I).LE.FHIGH.AND.FMODE(I).GE. FLOW) GO TO 708
      WRITE (IO2,104) I
      ITER=1
  708 CONTINUE
      IF (ITER .EQ. 1 ) GO TO 9960
C     GENERATES CUMULATIVE BETA TABLES FOR NINE BETA EQUATIONS
       DO 70 M = 1,9
      J=0
      NTABLE = M
      DELTA= 0.0078125
      A= 0.0
      BB=DELTA
       DO 70 J = 1,128
      CALL EVLINT(SFX,A,BB,FF)
      CUM(M,J)=FF
      XTABLE(J)=BB
      BB=BB+DELTA
   70  CONTINUE
      IHIST=1
 9908  CALL SAMPLE
C
C    SAMPLE RETURNS NSETS VALUES OF FDATA FOR USE AS INPUT TO THE COST MODEL
C
C    THE COST MODEL IS INSERTED HERE
      OUTPUT(IHIST)=FDATA(1)+FDATA(2)+FDATA(3)+FDATA(4)+FDATA(5)+
     *FDATA(6)+FDATA(7)
C
C   *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *
C   *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *
C
      IF (IHIST.GE.ITER) GO TO 9960
       IHIST = IHIST+1
       GO TO 9908
 9960 CONTINUE
      WRITE (IO3) ITER,NI,NPAGES,NSETS
      WRITE (IO3) (OUTPUT(K),K=1,ITER)
```

```
      BACKSPACE 1
      BACKSPACE 10
      CALL EXIT
  996 WRITE (102,100) I
      ITER=1
      GO TO 709
   50 FORMAT(I3,1X,I5)
   51 FORMAT(3(E10.3,1X),I1)
  101 FORMAT (46H DATA ERROR --- DISTRIBUTION TYPE FOR DATA SET,I4,1X,24
     *H1S ZERO OR NOT SPECIFIED)
  103 FORMAT (32X,3E10.2,I1)
  100 FORMAT(55H DATA ERROR -- LOW GREATER THAN HIGH IN DATA SET NUMBER,
     *I3)
  104 FORMAT(56H DATA ERROR -- MODE NOT BETWEEN HIGH AND LOW IN DATA SET
     *,I4)
      END
$IBFTC SSSSSX
      FUNCTION SFX(X)
      COMMON/NONAME/NTABLE,NSETS,IRANDM,XTABLE(128),FDATA(100),
     * FMODE(100),FRANGE(100),NTYPE(100),ITER,NI,CUM(9,128)
C
C     THIS SUBROUTINE--USED BY PKLEQ-- DEFINES BETA EQUATION PARAMETERS
C
      DIMENSION CONSTB (3,9)
      DATA CONSTB /1.5,.5,5.1,1.35,1.35,10.66,.5,1.5,5.1,3.0,1.0,20.0,2
     * .75,2.75,95.5,1.0,3.0,20.0,4.5,1.5,72.5,4.0,4.0,630.,1.5,4.5,72.5
     * /
      SFX=CONSTB (3,NTABLE)*(X**CONSTB(1,NTABLE))*((1.0-X)**CONSTB(2,NT
     * ABLE))
      RETURN
      END
$IBFTC SAMPLE
      SUBROUTINE SAMPLE
C
C     GENERATES A MONTE CARLO VALUE FOR EACH INPUT PARAMETER
C
      COMMON/NONAME/NTABLE,NSETS,IRANDM,XTABLE(128),FDATA(100),
     * FMODE(100),FRANGE(100),NTYPE(100),ITER,NI,CUM(9,128)
      DATA JK/100043/
      DATA THOU/100000./
      REAL MODTYP
      DIMENSION MODTYP(9)
      DATA MODTYP/.75,.50,.25,.75,.50,.25,.75,.50,.25/
      DO 99 N=1,NSETS
      IF(NTYPE(N).LT.1.OR.NTYPE(N).GT.9)GO TO 151
      I=NTYPE(N)
      SMODE=MODTYP(I)
      M=NTYPE(N)
      GO TO 10
C
C     SINGLE VALUED INPUT
C
  151 FDATA(N)=FMODE(N)
      GO TO 99
   10 L=IRANDM
    3 L=78125*L
      L=L-(L/JK)*JK
      IF(L-100000)4,4,3
    4 XL=FLOAT(L)/THOU
    5 IRANDM=L
      J = 64
      DO 13 K=1,7
```

```
      IF(K.EQ.7) GO TO 14
      L = 6 - K
      IF(CUM(M,J).LT.XL)GO TO 11
      IF(CUM(M,J).GT.XL) GO TO 12
      GO TO 14
   11 J = J + 2**L
      GO TO 13
   12 J = J - 2**L
   13 CONTINUE
   14 SAMPLX = XTABLE(J)
      FDATA(N)=FMODE(N)+FRANGE(N)*(SAMPLX-SMODE)
   99 CONTINUE
  100 RETURN
      END
$IBFTC INTEGR
      SUBROUTINE EVLINT(F, A, B, Y )
      DIMENSION SUB(10), WGT(5)
      DATA                SUB/.130467360E-1,.674683170E-1,
     A .160295216, .283302303, .425562831, .986953264,
     B .932531683,      .839704784,      .716697697,      .574437169/
      DATA              WGT/.333356722E-1, .747256746E-1,
     A .109543181,      .134633359,      .1477621127
      DX=B-A
      Y=0.0
      DO 100 I=1,5
      X1=SUB(I   )*DX+A
      X2=SUB(I+5)*DX+A
  100 Y=Y+WGT(I)*(F(X1)+F(X2))
      Y=Y*DX
      RETURN
      END
$ENTRY          BETATP
   7664325
                                    +1.00E+04 +1.60E+04 +1.20E+046
                                    +1.90E+04 +2.80E+04 +2.10E+043
                                    +1.20E+04 +3.00E+04 +1.80E+043
                                    +1.00E+04 +1.40E+04 +1.10E+046
$IBSYS
$IBJOB COMPLT  NODECK
$IBFTC HSTPLT  NODECK
C     EXTRACT OF HISTO
      IO1=5
      IO2=6
      IO3=10
      DIMENSION NFREQ(100),XVAL(100),SMOOTH(100),OUTPUT(1000)
      INTEGER BLANK,X,DOT,APOS
      DATA BLANK,X,DOT,APOS/1H ,1HX,1H.,1H'/
      READ (IO3) ITER,NI,NPAGES,NSETS
      IF (ITER .EQ.1 ) STOP
      READ(IO3) (OUTPUT(K),K=1,ITER)
      ITER25=ITER/25
      KOR=25
C     CALCULATES MEAN, STANDARD DEVIATION, AND PREPARES HISTOGRAM
      IHOPE=0
  445 IHOPE=IHOPE+1
C     CALCULATE INTERVAL RANGE (RINT) FOR NI INTERVALS
      XMIN = OUTPUT(1)
      XMAX = OUTPUT(1)
       DO 91 K = 1,ITER
      IF(OUTPUT(K).LT.XMIN) XMIN=OUTPUT(K)
      IF(OUTPUT(K).GT.XMAX) XMAX=OUTPUT(K)
```

```
   91  CONTINUE
       RINT = (XMAX-XMIN)/FLOAT(NI)
C      CALCULATE MEAN VALUE (XVAL) FOR EACH INTERVAL
       XVAL(1)=XMIN + RINT/2.
        DO 90 N = 2,NI
   90  XVAL(N)=XVAL(N-1) + RINT
C      DETERMINE DATA FREQUENCY FOR EACH INTERVAL
       IF(IHOPE.EQ.2) GO TO 93
       DO 20 N =1,NI
   20  NFREQ(N)=0
        DO 92  K = 1,ITER
       DO 21  N = 1,NI
       XLIM=XVAL(N)+RINT/2.
       IF(XLIM.GE.OUTPUT(K)) GO TO 92
   21  CONTINUE
       N = NI
   92  NFREQ(N)=NFREQ(N)+1
C      CALCULATE MEAN VALUE (XMEAN)
   93  SUMX = 0.
       DO 30  K = 1,ITER
   30  SUMX=SUMX+OUTPUT(K)
       XMEAN = SUMX/FLOAT(ITER)
C      CALCULATE STANDARD DEVIATION (XSDEV)
       SUMSQ = 0.
       DO 40  K = 1,ITER
   40  SUMSQ=SUMSQ+(OUTPUT(K)-XMEAN)**2
       IF (ITER.NE.1) GO TO 3210
       XSDEV=0.
       GO TO 3200
 3210  XVAR = SUMSQ/(FLOAT(ITER)-1.)
       XSDEV = SQRT(XVAR)
 3200  CONTINUE
C
C     PLOTTING ROUTINE
C
       DIMENSION LABELX(15),LABELY(10),LINE(132)
C
C     DETERMINES WIDTH OF PLOTTING INTERVAL
C
       INTWID=50/NI
       IF(INTWID.EQ.0) INTWID=1
C
C     DESIGNS X AXIS SCALE
C
       XRANGE=XMAX-XMIN
       IFACTR=0
       XINCR=XRANGE/10.
       IF(XINCR.LT.1.) GO TO 498
  400  IF(XINCR.GE.1..AND.XINCR.LE.10.) GO TO 410
       XINCR=XINCR/10.
       IFACTR=IFACTR+1
       GO TO 400
  410  INCRX=XINCR
       IF(INCRX.EQ.3) INCRX=2
       IF(INCRX.GT.3.AND.INCRX.LE.7) INCRX=5
       IF(INCRX.EQ.8.OR.INCRX.EQ.9) INCRX=10
       IF(IFACTR.EQ.0) GO TO 420
       DO 420 I=1,IFACTR
       INCRX=10*INCRX
  420  CONTINUE
```

```
      IFACTR=0
      SCALOW=XMIN
  425 IF(ABS(SCALOW).GE.0. .AND. ABS(SCALOW).LE.10.) GO TO 430
      SCALOW=SCALOW/10.
      IFACTR=IFACTR+1
      GO TO 425
  430 LOSCAL=SCALOW
      IF(IFACTR.EQ.0) GO TO 440
      DO 440 I=1,IFACTR
      LOSCAL=10*LOSCAL
  440 CONTINUE
  431 IF(FLOAT(LOSCAL+INCRX) .GE. XMIN) GO TO 432
      LOSCAL=LOSCAL+INCRX
      GO TO 431
  432 CONTINUE
      LABELX(1)=LOSCAL+INCRX
      DO 450 I=2,15
  450 LABELX(I)=LABELX(I-1)+INCRX
C
C     LABELX NOW CONTAINS 15 INCREMENTAL VALUES TO BE USED
C        FOR SCALE MARKINGS ALONG THE X AXIS
C
C     DESIGN Y AXIS SCALE
C
      MAXY=(ITER*5)/NI
      INCRY=MAXY/10
      LABELY(1)=INCRY
      DO 460 I=2,10
  460 LABELY(I)=LABELY(I-1)+INCRY
C
C     LABELY NOW CONTAINS 10 INCREMENTAL VALUES TO BE USED
C         FOR SCALE MARKINGS ALONG THE Y AXIS
C
C
C     PRINT Y SCALE
C
      WRITE (IO2,402)(LABELY(I),I=1,10)
      DO 469 I=16,116,10
  469 LINE(I)=DOT
      DO 481 I=1,15
  481 LINE(I)=BLANK
      DO 482 I=17,115,2
  482 LINE(I)=BLANK
      DO 483 ISTART=18,108,10
      ISTOP=ISTART+6
      DO 483 I=ISTART,ISTOP,2
  483 LINE(I)=APOS
      WRITE(IO2,403)(LINE(I),I=1,116)
      WRITE(IO2,404)XMIN
C
C     BEGIN PLOTTING DATA POINTS AND LABELING X AXIS
C
      DX = XRANGE/ (FLOAT(INTWID)*FLOAT(NI))
      DY = FLOAT(MAXY)/100.
      XVALU=XMIN
      LAB=1
      DO 499 K=1,NI
      DO 499 J=1,INTWID
      YVALU=0.
      I=1
  477 IF (FLOAT(NFREQ(K)).LE.YVALU) GO TO 475
```

```
      LINE(I)=BLANK
      I=I+1
      YVALU=YVALU+DY
      GO TO 477
  475 LINE(I)=X
      XVALU=XVALU+DX
C
C     PRINT X VALUE IF APPLICABLE
C
      IF(FLOAT(LABELX(LAB)) .GT. XVALU) GO TO 470
C
C     PRINT XMEAN IF IT OCCURS IN THIS INTERVAL
C
      IF(XMEAN.LT.XVALU .OR. XMEAN.GE.(XVALU+DX)) GO TO 471
      GO TO 472
  470 IF (XMEAN.LT.XVALU .OR. XMEAN.GE.(XVALU+DX)) GO TO 473
      GO TO 474
  471 WRITE(IO2,405)LABELX(LAB),(LINE(II),II=1,I)
      LAB=LAB+1
      GO TO 499
  472 WRITE(IO2,406)LABELX(LAB),XMEAN,(LINE(II),II=12,I)
      LAB=LAB+1
      GO TO 499
  473 WRITE (IO2,407) (LINE(II),II=1,I)
      GO TO 499
  474 WRITE(IO2,408)XMEAN,(LINE(II),II=12,I)
  499 CONTINUE
      WRITE(IO2,411)XMAX
      WRITE(IO2,412)XSDEV
  412 FORMAT (///,9H STD DEV=,F10.3)
      GO TO 99
  498 WRITE(IO2,409)
   99 CONTINUE
      IF(IHOPE.NE.1) CALL EXIT
      CALL POLYFT (XVAL,NFREQ,NI,KOR,SMOOTH)
      DO 505 I=1,NI
  505 NFREQ(I)=SMOOTH(I) + .500
      GO TO 445
  402 FORMAT(1H1,12X,4H 000,10(6X,I4))
  403 FORMAT(116A1/)
  404 FORMAT(6HXMIN =,F10.3//)
  405 FORMAT(I6,9X,100A1)
  406 FORMAT(I6,3X,7HXMEAN =,F10.3,88A1)
  407 FORMAT(15X,100A1)
  408 FORMAT(9X,7HXMEAN =,F10.3,88A1)
  409 FORMAT (79HOERROR MESSAGE --- RANGE OF TOTAL COST ESTIMATES IS LES
     *S THAN TEN. PLOT DELETED )
  411 FORMAT(6HXMAX =,F10.3)
      END
$IBFTC FTKURV
      SUBROUTINE POLYFT(XVAL,NFREQ,NI,KOR,SMOOTH)
      IO1=5
      IO2=6
      IO3=10
C                                                              POLY0010
C     DIMENSION  FOR ARGUMENTS                                 POLY0020
      DIMENSION XVAL(100),NFREQ(100),A(625),C(25),SMOOTH(100)
C                                                              POLY0040
C     DIMENSION  FOR SELF-GENERATED VALUES                     POLY0050
      DIMENSION  SUMX(100),SMYX( 50), AMEANX( 50)              POLY0060
```

```
      PTS=FLOAT(NI)
      KTOR= 2*KOR                                                    POLY0090
C                                                                    POLY0100
C     INITIALIZATION                                                 POLY0110
      DO  1  I=1,KOR                                                 POLY0120
      SMYX(I)= 0.0                                                   POLY0130
    1 AMEANX(I)= 0.0                                                 POLY0140
      DO  2  I=1,KTOR                                                POLY0150
    2 SUMX(I)= 0.0                                                   POLY0160
      SUMY= 0.0                                                      POLY0170
C                                                                    POLY0180
C     NORMALIZATION WITH RESPECT TO XMAX                             POLY0190
      XVAL1=XVAL(1)
      DO 707 NOJ=1,NI
  707 XVAL(NOJ)=XVAL(NOJ)-XVAL1
      XMAX=XVAL(NI)
      DO 102 I=1,NI
  102 XVAL(I)=XVAL(I)/XMAX
C                                                                    POLY0270
C                                                                    POLY0280
C     FORMULATION OF NORMAL EQUATIONS                                POLY0290
      DO  3 J=1,KTOR                                                 POLY0300
      DO 3 I=1,NI
    3 SUMX(J)=SUMX(J)+XVAL(I)**J
      DO 4 I=1,NI
    4 SUMY=SUMY+FLOAT(NFREQ(I))
      AMEANY= SUMY/PTS                                               POLY0350
      DO  6 J=1,KOR                                                  POLY0360
      AMEANX(J)= SUMX (J)/PTS                                        POLY0370
      DO 6 I=1,NI
    6 SMYX(J)=SMYX(J)+FLOAT(NFREQ(I))*XVAL(I)**J
      DO  8 I=1,KOR                                                  POLY0400
      C(I)= SMYX(I) -PTS*AMEANX(I)*AMEANY                            POLY0410
      DO  8 J=1,KOR                                                  POLY0420
      K= I+J                                                         POLY0430
      IJ =(J-1)*KOR +I                                               POLY0440
    8 A(IJ) =SUMX(K) -PTS* AMEANX(I)*AMEANX(J)                       POLY0450
C                                                                    POLY0460
C     CROUT'S  REDUCTION  METHOD                                     POLY0470
      DO  11 I=2,KOR                                                 POLY0480
      I1I=(I-1)*KOR + 1                                              POLY0490
   11 A(I1I) = A(I1I)/ A(1)                                          POLY0500
      DO  12 J=2,KOR                                                 POLY0510
      KM= J-1                                                        POLY0520
      DO 14 I=J,KOR                                                  POLY0530
      AP1= 0.0                                                       POLY0540
      DO114 K=1,KM                                                   POLY0550
      IK =(K-1)* KOR+ I                                              POLY0560
      KJ =(J-1)* KOR+ K                                              POLY0570
  114 AP1 = AP1 + A(IK) *A(KJ)                                       POLY0580
      IJ =(J-1)* KOR+ I                                              POLY0590
   14 A(IJ) = A(IJ) -AP1                                             POLY0600
      JP= J+1                                                        POLY0610
      IF (JP- KOR) 444, 444, 445                                     POLY0620
  444 DO 16 I=JP,KOR                                                 POLY0630
      AP1= 0.0                                                       POLY0640
      DO116 K=1,KM                                                   POLY0650
      JK =(K-1)* KOR + J                                             POLY0660
      KI =(I-1)* KOR + K                                             POLY0670
  116 AP1 =AP1 +A(JK) *A(KI)                                         POLY0680
      JI =(I-1)* KOR + J                                             POLY0690
```

```
      JJ =(J-1)* KOR + J                                                POLY0700
   16 A(JI) = (A(JI) -AP1)/A(JJ)                                        POLY0710
  445 D 1MY= 0.0                                                        POLY0720
   12 CONTINUE                                                          POLY0730
      C(1) =C(1)/A(1)                                                   POLY0740
      DO  18 I=2,KOR                                                    POLY0750
      AP1= 0.0                                                          POLY0760
      IM=I-1                                                            POLY0770
      DO 118 K=1,IM                                                     POLY0780
      IK =(K-1)* KOR + I                                                POLY0790
  118 AP1=AP1 +A(IK) *C(K)                                              POLY0800
      II =(I-1)* KOR + I                                                POLY0810
   18 C(I) =(C(I)- AP1) / A(II)                                         POLY0820
      KORM= KOR-1                                                       POLY0830
      IF (KORM) 122, 123,122                                            POLY0840
  122 DO  21 I=1,KORM                                                   POLY0850
      AP1= 0.0                                                          POLY0860
      M= KOR-I                                                          POLY0870
      MP= M+1                                                           POLY0880
      DO 121 K=MP,KOR                                                   POLY0890
      MK =(K-1)* KOR +M                                                 POLY0900
  121 AP1 =AP1 + A(MK)* C(K)                                            POLY0910
   21 C(M) =C(M) -AP1                                                   POLY0920
  123 AP1= 0.0                                                          POLY0930
      DO 24  I=1,KOR                                                    POLY0940
   24 AP1 =AP1 +AMEANX(I) *C(I)                                         POLY0950
      CO  = AMEANY -AP1                                                 POLY0960
C                                                                       POLY0970
  778 SRES = 0.0                                                        POLY1030
      WRITE(IO2,29)
   29 FORMAT (60H1          COST                FREQ        SMOOTHED
     * RES/43H          INTERVAL                            FREQ)
      DO 77 I=1,NI
      SMOOTH(I)=CO
      DO  27 J=1,KOR                                                    POLY1090
   27 SMOOTH(I)=SMOOTH(I)+C(J)*XVAL(I)**J
      RES=FLOAT(NFREQ(I))-SMOOTH(I)
C                                                                       POLY1130
C     DENORMALIZATION WITH RESPECT TO XMAX                              POLY1140
      XVAL(I)=XVAL(I)*XMAX
      XVAL(I)=XVAL(I)+XVAL1
   30 FORMAT( 4F15.4)                                                   POLY1160
      FREQ=FLOAT(NFREQ(I))
   77 WRITE(IO2,30)XVAL(I),FREQ,SMOOTH(I),RES
      RETURN                                                            POLY1270
      END                                                               POLY1280
$ENTRY        HSTPLT
$CLOSE        S.SU10
```

REFERENCES

1. S. Sobel, A Computerized Technique to Express Uncertainty in Advanced System Cost Estimates, TM-3728, The Mitre Corporation, 1963.
2. David B. Hertz, "Risk Analysis in Capital Investment," Harvard Business Review, 42(1): (Jan–Feb 64).
3. Paul F. Dienemann, Estimating Cost Uncertainty Using Monte Carlo Techniques, RM-4854-PR, The RAND Corporation, 1966.
4. W. D. Lamb, A Technique for Probability Assignment in Decision Analysis, General Electric Major Appliance Division, 1967.
5. A. M. Mood and F. A. Graybill, Introduction to the Theory of Statistics, 2d ed., McGraw-Hill Book Co., New York, 1963.
6. Waloddi Weibull, "A Statistical Distribution Function of Wide Applicability," Journal of Applied Mechanics, Sep 51.
7. "IBM 7040/7044 Operating System (16/32K) Fortran IV Language," File 7040-25, Form L-28-6329-3.

RAC DISTRIBUTION LIST A

1 Sep 68

(Less these deletions: O17, W1, W2, W3, W4, W5, E20 and E21)

Address code	Agency	Number of copies
	DEPARTMENT OF DEFENSE	
D3	Armed Forces Staff College	1
A9	Assistant Secretary of Defense, Systems Analysis	2
A3	Assistant Secretary of Defense, International Security Affairs	1
C2	Defense Atomic Support Agency	1
C4	Defense Communications Agency (DCA)	1
C6	Defense Document Center (DDC)	20
C8	Defense Intelligence Agency, Medical Branch, Environment Division	1
C9	Defense Intelligence Agency (DIA SA-2B)	1
A2	Director of Defense Research and Engineering	1
C1	Advanced Research Projects Agency	1
B5	Weapons Systems Evaluation Group	1
D2	Industrial College of the Armed Forces	1
F3	Joint Support Command	1
C3	National Security Agency	1
D1	National War College	1
	JOINT CHIEFS OF STAFF	
B2	Joint Chiefs of Staff	1
B4	Joint War Game Control Group	1
B6	Special Study Group	1
	JOINT COMMANDS	
L5	Commander in Chief, Alaska (CINCAL)	1
L12	Commander in Chief, Europe (CINCEUCOM)	1
L6	Commander in Chief, Pacific (CINCPAC)	1
L7	Supreme Headquarters, Allied Powers, Europe (USNMR)	1
L16	US Strike Command, MacDill Air Force Base	1
	HEADQUARTERS, DEPARTMENT OF THE ARMY	
	Secretary of the Army	
E2	Office, Under Secretary of the Army, Operations Research	1
E4	Assistant Secretary of the Army (R&D)	1
M26	Data Systems Office, Research Division, Army Materiel Command	1
	CHIEF OF STAFF, US ARMY	
E33	Assistant Vice Chief of Staff of Army	5
	DEPARTMENT OF THE ARMY	
E23	Assistant Chief of Staff for Force Development	1
R11	Army Concepts Team, Vietnam	1
O24	Army War College	1
E10	Assistant Chief of Staff, Intelligence	2
O9	Army Intelligence School, Ft Holabird	1
E15	The Army Library, Attn: ASDIRS	1
E19	Chief of Communications-Electronics	1
O27	Academic Computer Center, West Point, N. Y.	1
E17	Chief of Engineers	1
E20	Director of Transportation	1
E27	Chief of Research and Development	1
R4	US Army Behavioral Science Research Laboratory	1
E21	Chief of Support Services	1
E14	Comptroller	1
E9	Deputy Chief of Staff for Logistics, Doctrine Branch	1
E8	Deputy Chief of Staff for Military Operations	2(4)[a]
P9	Army Strategic and Tactical Analysis Group	1
E7	Deputy Chief of Staff for Personnel	1
E18	Surgeon General	1
K11	Defense Logistics Studies Information Exchange, US Army Logistics Management Center, Ft Lee	2
	ARMIES, CONUS AND OVERSEAS	
H2	First	1
H3	Third	1
H4	Fourth	1
H5	Fifth	1
H6	Sixth	1
H8	Eighth	1

[a] Send 4 copies of RAC Strategic Studies Department documents, but only 2 copies of all other RAC documents.

DOCUMENT CONTROL DATA - R&D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY (Corporate author) Research Analysis Corporation McLean, Virginia 22101	2a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED 2b. GROUP

3. REPORT TITLE

A MONTE CARLO SIMULATION APPROACH TO COST-UNCERTAINTY ANALYSIS

4. DESCRIPTIVE NOTES (Type of report and inclusive dates)

Technical Paper

5. AUTHOR(S) (First name, middle initial, last name)

Donald F. Schaefer, Frank J. Husic, and Michael F. Gutowski

6. REPORT DATE	7a. TOTAL NO. OF PAGES	7b. NO. OF REFS
March 1969	60	7

8a. CONTRACT OR GRANT NO.	9a. ORIGINATOR'S REPORT NUMBER(S)
DA 44-188-ARO-1 b. PROJECT NO. 009.135 c. d.	RAC-TP-349 9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report)

10. DISTRIBUTION STATEMENT

This document is subject to special export controls and each transmittal to foreign governments or foreign nationals may be made only with prior approval of the Comptroller of the Army (ATTN: COMPT-CA(R)), Headquarters, Department of the Army, Washington, D. C., 20310.

11. SUPPLEMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY
	Office of the Comptroller of the Army

13. ABSTRACT

An important aspect of cost research is the measurement of the uncertainty inherent in the projection of system cost. Approaches to this problem have in the past centered on the decision maker's intuition or in sensitivity analysis. Only recently have approaches utilizing such tools as statistical decision theory and probability theory been formulated.

This study focuses on the Monte Carlo simulation approach to uncertainty in cost analysis. This approach requires:

(a) Expression of input estimates as probability distributions reflecting uncertainty.

(b) Cost equations pertinent to a particular model.

The Monte Carlo simulation approach then generates:

(a) The frequency distribution for system cost.

(b) Statistical measures that illustrate the nature and magnitude of system cost uncertainty.

Two models are developed, the Beta model and the Weibull model, each of which reflects a particular distribution form for the inputs. The relative costs and advantages of each model are compared.

A user's guide to the program and complete program listings are presented in an appendix.

DD FORM 1 NOV 68 1473

14. KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
Beta distribution cost uncertainty Monte Carlo simulation sensitivity analysis Weibull distribution						